Quantitative Automata and Logic for Pictures and Data Words

Von der Fakultät für Mathematik und Informatik
der Universität Leipzig
angenommene

D I S S E R T A T I O N

zur Erlangung des akademischen Grades

Doctor rerum naturalium
(Dr. rer. nat.)

im Fachgebiet
Informatik

vorgelegt
von M. Sc. Parvaneh Babari
geboren am 21. Juni 1985 in Teheran, Iran.

Die Annahme der Dissertation wurde empfohlen von:

1. Professor Dr. Manfred Droste (Universität Leipzig)
2. Dr. habil. Benedikt Bollig (Universität Cachan, Frankreich)

Die Verleihung des akademischen Grades erfolgt mit Bestehen
der Verteidigung am 03.03.2017 mit dem Gesamtprädikat
magna cum laude

Acknowledgements

First of all, I would like to thank my advisors Prof. Dr. Manfred Droste and Prof. Dr. Heiko Vogler for encouraging my research and for providing me with the opportunity to grow as a researcher. Their advice on both research and on my career have been priceless. I especially want to thank my advisor on the spot, Prof. Dr. Manfred Droste, whose support and guidance made my thesis work possible. I am very grateful for his patience, motivation, enthusiasm, and immense knowledge in the theory of weighted automata and weighted logic.

As a member of the Graduiertenkolleg 1763 "Quantitative logics and automata", I am very grateful for being financially supported by *Deutsche Forschungsgemeinschaft* (DFG) during my research.

I would like to thank Prof. Dr. Nicole Schweikardt for the valuable and insightful discussions and interesting ideas concerning Chapter 2 of this thesis. Thanks to Karin Quaas and Vitaly Perevoshchikov, colleagues and good friends of mine, who have been helpful in providing advice many times during my research. I enjoyed working with them in the last few years, and I have learned a lot from them. I would like to say thanks again to Vitaly for reading a preliminary version of this thesis and providing me with his nice comments. I am grateful to Mahsa Shirmohammadi, who introduced me to the topic of synchronization and for the valuable discussions concerning our joint work, partly included in Chapter 4 of this thesis.

I would like to thank my friend and former colleague of mine Claudia Carapelle, for her great friendship and moral support during the last few years in Germany.

Finally, I would like to thank my family for their love, support, encouragement and patience during the last few years being away from them. Words cannot express how grateful I am to all of you. I would like to dedicate my thesis to my father, whose wish was always my success. I will always feel him right by my side, in every step of my life!

Contents

Introduction

Mathematical logic and automata theory are two scientific disciplines with a close relationship that is not only fundamental for many theoretical results but also forms the basis of a coherent methodology for the verification and synthesis of computing systems. This connection goes back to a much longer history in the 1960s, through the fundamental work of Büchi-Elgot-Trakhtenbrot [25, 51], which shows the expressive equivalence of automata and logical systems such as monadic second-order logic on finite and infinite words. This allowed the handling of specifications (where global system properties are stated), and implementations (which involve the definition of the local steps in order to satisfy the global goals laid out in the specifications) in a single framework. This connection has been extended to and well-investigated for many other structures such as trees [96], finite pictures [64], timed words [103] and data words [22].

For many computer science applications, however, quantitative phenomena need to be modelled, as well. Examples are vagueness and uncertainty of a statement, length of time periods, spatial information, and resource consumption. Weighted automata, introduced by Schützenberger [90], are prominent models for quantitative aspects of systems. This soon led to a flourishing theory cf. [42, 50, 73, 88]. The framework of weighted monadic second-order logic over words was first introduced by Droste and Gastin [41]. They gave a characterization of quantitative behavior of weighted finite automata, as semantics of monadic second-order sentences within their logic. Meanwhile, the idea of weighted logics was also applied to devices recognizing more general structures such as weighted tree automata [48], weighted automata on infinite words [46] or traces [80]. The *main goal* of this thesis is to give logical characterizations for weighted automata models on pictures and data words as well as for Büchi-tiling systems in the spirit of the classical Büchi-Elgot theorem. As the second goal, we deal with *synchronizing problem for data words*. We will introduce this problem later.

A *picture* is a finite rectangular array of elements from a finite alphabet. The theory of picture languages is motivated by problems arising from image processing and pattern recognition [60, 81, 95, 102], and also plays a role in the theory of cellular automata and other devices of parallel computing [76, 93]. The family of recognizable picture languages was defined and characterized in the nineties by many different devices [61, 63, 67, 69]. Several research groups obtained a description of recognizable picture languages in terms of automata, sets of tiles, rational operations, and existential monadic second-order logic [62, 64, 67, 75]. A semiring-weighted (quadrapolic) picture automaton model was first introduced by Bozapalidis and Grammatikopoulou [24]. The behavior of such a picture automaton is a *picture series* which maps pictures over an arbitrary alphabet to elements of the semiring. In 2006, Fichtner provided a notion of a weighted MSO logic over pictures [52, 53, 54]. For commutative semirings, she proved that the class of picture series defined by sentences of the weighted logics coincides with those computed by weighted picture automata [52].

In Chapter 1, we introduce a new weight structure called *picture valuation monoids*. Then, we introduce *weighted two-dimensional on-line tessellation automata* (W2OTA) taking weights from picture valuation monoids. This new weighted automaton model operating on pictures enables us to model several application examples, e.g. the average light of a picture (interpreting the symbols of the alphabet as different levels of light). Note that such aspects can not be modelled with commutative semirings. Weighted automata over words computing objectives like average cost or long-time peak power consumption were introduced recently by Chatterjee, Doyen, and Henzinger [28, 29, 30, 31].

As one of the main results in Chapter 1, we prove a Nivat-like theorem for recognizable picture series, i.e., for the behaviors of W2OTA. Nivat's Theorem is a fundamental characterization of rational transductions, which provides a connection between rational transductions and rational languages; see [43] for a version of this result for semiring-weighted automata on words. Following the approach applied in [45] for recognizable quantitative timed languages, we show that recognizable picture series can be obtained precisely as projections of particularly simple unambiguously recognizable series restricted to unambiguously recognizable picture languages. In addition, we show that if the given picture valuation monoid is idempotent, then we do not need unambiguity of the underlying picture language.

Successively in Chapter 1, combining the ideas from [19, 44, 45, 52], we define a new weighted MSO logic, which can model average density of pictures. While in [41, 52], disjunction and existential quantification were interpreted by the sum, and the semantics of both conjunction and universal quantification were defined by the product operation of the semiring, in our logic the semantics of universal quantification will be interpreted by a picture valuation function. As the most substantial result of this chapter, we prove that, under suitable assumptions on the underlying picture valuation monoid, our weighted automaton device and a suitable fragment of weighted MSO logic are expressively equivalent. To reach this goal, we define four different fragments: In all cases the application of universal first order quantification is restricted to the class of almost boolean first-order formulas, which is the smallest class containing all constants from our weight structure, and all boolean first-order formulas and which is closed under conjunction and disjunction. In one case we also restrict the use of constants in the formula by allowing their occurrence only in the scope of the first-order universal quantifier. The other fragments differ in the application of conjunction. Considering these restrictions makes our results different from the ones in [52], in which the strong assumption of commutativity over semirings has been considered.

The notion of recognizability have also been studied for languages of $\omega\omega$-pictures [1, 35, 59, 65, 66, 98]. An $\omega\omega$-picture is indeed considered as an ω-word generalized to two dimensions. However, a logical characterization for the class of recognizable $\omega\omega$-picture languages has not been obtained so far. Due to the technical difficulties caused by infinity in two dimensions, the approach in [64] for logical characterization of finite picture languages cannot be easily extended to the class $\omega\omega$-picture languages. Our next goal is to find a new class of infinite picture languages for which obtaining a logical characterization is possible.

In Chapter 2, we study *$+\omega$-pictures*, i.e., pictures that have a finite number of rows and an infinite number of columns. As a motivation for studying these pictures, consider for example the potentially infinite streams of taped videos captured by digital security cameras. To the best of our knowledge, languages of $+\omega$-pictures have not been studied before. To obtain a notion of recognizability of languages of $+\omega$-pictures, we introduce *Büchi-tiling systems*. They extend the classical tiling systems [63] with a Büchi acceptance condition. This way, Büchi-tiling recognizable $+\omega$-picture languages can be viewed as a natural generalization of ω-regular

languages. We show that the class of Büchi-tiling recognizable $+\omega$-picture languages has the same closure properties as the class of tiling recognizable languages of finite pictures [64], i.e., it is closed under projection, union, and intersection, but not under complementation. We characterize the family of $+\omega$-picture languages by generalized Büchi-tiling systems and by an *extension* of existential monadic second-order logic with quantification of infinite sets (EMSO$^\infty$). Furthermore, we show that many results and techniques that have been developed for the tiling recognizable languages of finite pictures or for the ω-regular languges recognized by Büchi-automata can be transferred to Büchi-tiling recognizable $+\omega$-picture languages. However, using combinatorial arguments, we show that the well-known Büchi characterization theorem (stating that the ω-regular languges are unions of finitely many languages of the form $L_1 \cdot L_2^\omega$, for regular languages L_1 and L_2) does not carry over to the Büchi-tiling recognizable languages of $+\omega$-pictures.

Our focus in the rest of this thesis is on data words. A *data word* is a sequence of pairs where the first element is taken from a finite alphabet and the second element is taken from an infinite data domain such as natural numbers or ASCII strings. In recent years, data words have become an active subject of research due to its numerous applications in querying and reasoning about data models with complex structural properties, in XML, in particular in static analysis of logic and automata-based XML specifications, and lately also in graph databases [6, 12, 17, 56]. *Register automata* introduced by Kaminski and Francez [71] provide a widely studied model for reasoning on data words. These automata can be considered as classical nondeterministic finite automata equipped with a finite set of registers which are used to store data in order to compare it with some data in the future. This enables them to handle parameters like user names, passwords, identifiers of connections, etc., in a fashion similar to, and slightly more expressive than, the class of *data-independent* systems. This model served as a basis for the study of various automata models and logics on data words and trees [16, 22, 36, 37, 38, 55, 58, 77, 82, 87]. Classical *timed automata* of Alur and Dill [2] for real-time systems are another example of automata on data (timed) words.

In Chapter 3, for quantitative reasoning on data words, we introduce *weighted register automata* over *commutative data semirings* equipped with a collection of binary data functions in the spirit of the classical theory of weighted automata [42]. Whereas in the models of register automata known from the literature data are usually compared with respect to equality or

a linear order, here we allow data comparison by means of an arbitrary collection of binary data relations. This approach permits easily to incorporate timed automata [2] and weighted timed automata [5, 86] into our framework. Moreover, this approach gives rise to further investigations of models for data processing, e.g., data comparison with an approximation error.

We introduce semiring-weighted existential MSO logic on data words equipped with binary data functions. In order to model data comparison, we use data predicates in the spirit of relative distance predicates (Wilke [103]). Our goal is to prove the expressive equivalence of this new logic with weighted register automata. However, to reach this goal we faced the following difficulties: The unrestricted use of binary data functions and weighted universal quantifiers goes beyond recognizability even for very simple formulas. In addition, register automata are neither determinizable nor closed under complement. In order to overcome these problems, we obtain a suitable fragment of our weighted existential MSO logic by restricting the use of the weighted universal quantifier to formulas without weighted quantifiers and by restricting the use of data functions to an intuitively defined logical operator. In our main result, we state that this restricted weighted EMSO logic is equivalent to weighted register automata. The existing proof techniques and ideas for weighted logic cannot be applied in our setting, since register automata are not closed under complement and we need a new construction to deal with binary data functions. For this purpose, we introduce a determinizable class of unweighted register automata, called *visibly register automata*. This determinizable class of register automata could be also of independent interest. Moreover, we introduce a new normalization technique for the binary data functions, in order to provide a translation of formulas with weighted universal quantifiers into weighted register automata. With this, we achieve our goal; moreover, our construction of weighted register automata equivalent to a given weighted MSO formula is effective. To the best of our knowledge, quantitative extensions of register automata have not been studied before.

In the final chapter of this thesis, we deal with a slightly different and more practical problem. We consider *synchronizing data words in register automata*. Synchronizing words for finite state automata have been studied since the seventieth, see [26, 78, 84, 101]; a synchronizing word w drives the automaton from an unknown or unobservable state to a specific state q_w that only depends on w. Synchronizing words have applications in planning, control of discrete event systems, biocomputing,

and robotics [14, 40, 101]. Over the past few years, this classical notion has sparked renewed interest and has been generalized to games on transition systems [72, 74, 91], and to infinite-state systems [32, 39], which are more relevant for modelling complex and realistic systems such as distributed data networks or real-time embedded systems. These studies have inspired an elegant extension of temporal logics to capture the synchronizing properties [27]; the proposed logic is more expressive than classical computation tree logic. In Chapter 4, we investigate the problem of synchronizing data words in register automata. We introduce a very general register automata in Chapter 3; however, register automata come in different variants, e.g., one-way vs. two-ways, deterministic vs. non-deterministic, alternating vs. nonalternating. For alternating register automata, classical decision problems like non-emptiness, universality and language inclusion are undecidable. In pursuit of good feasability, in Chapter 4, we only focus on the class of one-way register automata without alternation; they have a decidable non-emptiness problem [71], and the subclass of nondeterministic register automata with a single register has a decidable non-universality problem [36].

Semantically, a register automaton defines an infinite-state system, due to the unbounded domain for data stored in registers. Synchronizing words were introduced for infinite-state systems with infinite branching in [39, 91]; in particular, the notion of synchronizing words has been motivated and studied for weighted automata and timed automata. In some infinite-state settings such as nested word automata (or equivalently visibly pushdown automata), finding the right definition of synchronizing words is however more challenging [32]. We define the synchronizing problem for register automata along the suggested framework in [39, 91]: Given a register automaton $\mathcal{A}$, does there exist a data word w that brings each of the infinitely many states of $\mathcal{A}$ to some specific state (depending only on w)? Such a data word is called a *synchronizing data word*.

The problem of finding synchronizing data words for register automata imposes new challenges in the area of synchronization. It is natural to ask how many distinct data are necessary and sufficient to synchronize a register automaton, which we refer to by the notion of *data efficiency* of synchronizing data words. We show that the data efficiency is polynomial in the number of registers for deterministic register automata (DRA), and is $\mathsf{Ackermann}(n)$ for nondeterministic register automata (NRA) where n is the number of states. Remarkably, data efficiency is tightly related to the complexity of deciding the existence of a synchronizing data word.

For DRA, we prove that for every register automaton $\mathcal{A}$ with k registers, if $\mathcal{A}$ has a synchronizing data word, then it also has a synchronizing data word with data efficiency *at most* $2k + 1$. We also provide a family of register automata with k registers, for which indeed a polynomial data efficiency (in the size of k) is necessary to synchronize. This bound is the base for an NLOGSPACE-algorithm for deciding the synchronizing problem for DRA with single register, and an PSPACE-algorithm for DRA with multiple registers. We prove a matching PSPACE lower bound by ideas carried over from timed settings [39].

For NRA, a reduction from the non-universality problem yields the undecidability of the synchronizing problem. For single-register NRA (1-NRA), we prove Ackermann-completeness of the problem by a novel construction proving that the synchronizing problem and the non-universality problem in 1-NRAs are polynomial-time interreducible. We believe that this technique is useful in studying synchronization in all nondeterministic settings, requiring careful analysis of the size of the construction.

Preliminary versions of the results of this thesis appeared in [8, 9, 10] and [11].

Chapter 1

Weighted Automata and Weighted MSO Logic on Pictures

Informally, a two-dimensional string is called a *picture* and is defined as a rectangular array of symbols taken from a finite alphabet. A *two-dimensional language* (or *picture language*) is a set of pictures. Picture languages have been intensively investigated by several research groups [61, 62, 63, 64, 67, 75]. In this chapter, we define *weighted two-dimensional on-line tessellation automata* (W2OTA) taking weights from a new weight structure called *picture valuation monoid*. This new weighted picture automaton model can be used to model several application examples, e.g. the average density of a picture. Such aspects could not be modelled by semiring weighted picture automaton models [52]. The behavior of this automaton model is a *picture series* mapping pictures over an alphabet to elements of a picture valuation monoid. As one of our main results, we prove a Nivat theorem for W2OTA. It shows that recognizable picture series can be obtained precisely as projections of particularly simple unambiguously recognizable series restricted to unambiguous recognizable picture languages. In addition, we introduce a weighted monadic second-order logic (WMSO) which can model average density of pictures. As the other main result, we show that W2OTA and a suitable fragment of our weighted MSO logic are expressively equivalent.

1.1 Picture Valuation Monoids and Weighted Picture Automata

Let $\mathbb{N} = \{0, 1, 2, \cdots\}$ be the set of natural numbers, and $\mathbb{N}_{\geqslant 1} := \mathbb{N}\backslash\{0\}$. We denote by $[n]$ simply the set $\{1, 2, \cdots, n\}$, for any $n \in \mathbb{N}_{\geqslant 1}$. An *alphabet* is a non-empty finite set. Let Σ be an alphabet. A *finite picture* over Σ is a finite rectangular array of elements of Σ. Formally, for $m, n \in \mathbb{N}_{\geq 1}$, a picture of *size* (m, n) over Σ is a mapping $p : [m] \times [n] \to \Sigma$. The number $\ell_v(p) := m$ of rows is called the *height* of p, and the number $\ell_h(p) := n$ of columns is called the *width* of p. As an example, consider the following picture of size $(4, 4)$ over the alphabet $\Sigma = \{0, 1\}$:

$$\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}$$

For $i \in [\ell_v(p)]$ and $j \in [\ell_h(p)]$ we write $p_{i,j} := p(i, j)$ to denote the letter of p in row i and column j. For $m, n \in \mathbb{N}_{\geqslant 1}$, we write $\Sigma^{m\times n}$ for the set of all pictures over Σ of size (m, n), and we let $\Sigma^{++} := \bigcup_{m,n\in\mathbb{N}_{\geqslant 1}} \Sigma^{m\times n}$ be the set of all finite pictures over Σ. A *picture language* L over Σ is a subset of Σ^{++}, i.e. $L \subseteq \Sigma^{++}$. For more details we refer the reader to [63, 64, 100].

Whereas in [52, 53, 54], *commutative semirings* were considered as the weight structure to define the behavior of weighted picture automata, here we define a new weight structure called *picture valuation monoid*. Note that valuation monoids were first introduced by Droste and Meinecke in [44] for weighted automata on finite and infinite words.

Definition 1.1. *A picture valuation monoid, or for short pv-monoid, is a tuple* $\mathbb{M} = (M, +, \mathrm{val}, \mathbf{0})$ *consisting of a commutative monoid* $(M, +, \mathbf{0})$ *and a valuation function* $\mathrm{val} : M^{++} \to M$ *with* $\mathrm{val}(d) = d$ *for all* $d \in M$ *and*

$$\mathrm{val}\left(\begin{pmatrix} d_{1,1} & d_{1,2} & \cdots & d_{1,n} \\ d_{2,1} & d_{2,2} & \cdots & d_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ d_{m,1} & d_{m,2} & \cdots & d_{m,n} \end{pmatrix}\right) = \mathbf{0},$$

whenever $d_{i,j} = \mathbf{0}$ *for some* i *and* j*, for* $d_{1,1}, ...d_{m,n} \in M$*. We say that* $\mathbb{M}$ *is idempotent if* $+$ *is idempotent, i.e.,* $d + d = d$ *for all* $d \in M$*.*

For simplicity of notation in the sequel we use

$$\mathrm{val}((d_{i,j})_{\substack{1\leqslant i\leqslant m \\ 1\leqslant j\leqslant n}}) \quad \text{or} \quad \mathrm{val}(d_{1,1}, ..., d_{1,n}, d_{2,1}, ..., d_{2,n}, ..., d_{m,1}, ..., d_{m,n})$$

instead of

$$\mathrm{val}\left(\begin{pmatrix} d_{1,1} & d_{1,2} & \cdots & d_{1,n} \\ d_{2,1} & d_{2,2} & \cdots & d_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ d_{m,1} & d_{m,2} & \cdots & d_{m,n} \end{pmatrix}\right).$$

Example 1.2. *Consider the pv-monoid* $(\mathbb{R} \cup \{-\infty\}, \sup, \mathrm{avg}, -\infty)$ *where* $\mathrm{avg}((d_{i,j})_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}}) = \frac{1}{m \times n} \sum_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}} d_{i,j}$*; note that the sum used in the definition of avg is the usual addition in* $\mathbb{R} \cup \{-\infty\}$*. Let* $B \subseteq [0,1]$ *be a finite set of values and let* $L \subseteq B^{++}$ *be any picture language over* B*. Consider the function* $\mathbb{L} : B^{++} \to \mathbb{R} \cup \{-\infty\}$ *defined for* $p \in B^{++}$ *by:*

$$\mathbb{L}(p) = \begin{cases} \frac{1}{m \times n} \sum_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}} p_{i,j} & \text{if } p \in L, \\ -\infty & \text{otherwise.} \end{cases}$$

We could interpret the values in B *as different levels of light. Then for each picture* p *in the language* L*, the function* S *provides the average value* $\mathbb{L}(p)$ *of light of* p*.*

Example 1.3. *Consider the pv-monoid* $(\mathbb{N} \cup \{\infty\}, \min, \mathrm{maj}, \infty)$ *where the valuation function is a majority function, i.e.,* $\mathrm{maj}(d_1, ..., d_n)$ *is the greatest value among all the values that occur most frequently, e.g.* $\mathrm{maj}(4, 6, 6, 6, 8, 8, 8, 12) = 8$*. Now let* B *be a finite set, interpreted for instance as a set of different colors, and consider the function* $\mathbb{L} : B^{++} \to \mathbb{N}$*, defined for* $p \in B^{++}$ *by:*

$$\mathbb{L}(p) = \mathrm{maj}\{\ell_h(q) \cdot \ell_v(q) \mid q \text{ is a monochrome subpicture of } p\}.$$

Then for each picture p*,* $\mathbb{L}$ *provides the largest area of a monochrome rectangle, enclosed as a subpicture within* p*, which can be found most frequently among all monochrome subpictures of* p*. We remark that the term* monochrome *is usually taken to mean that a picture contains only two colors, e.g. black and white.*

The family of recognizable picture languages has been characterized by many different devices [61, 62, 63, 64, 67, 75]. As one of these devices, one can consider 2-dimensional on-line tessellation automatoa (2OTA) introduced by Inoue and Nakamura [67, 69]:

Definition 1.4. *A 2-dimensional on-line tessellation automaton, for short* 2OTA*, over an alphabet* Σ *is a tuple* $\mathcal{A} = (Q, T, I, F)$ *consists of a finite set* Q *of states, a set of transitions* $T \subseteq Q \times Q \times \Sigma \times Q$*, sets of initial and final states* $I, F \subseteq Q$*, respectively.*

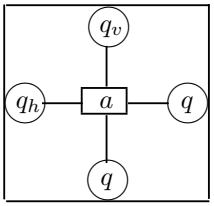


Figure 1.1: A transition $t = (q_h, q_v, a, q)$ of a W2OTA

For an alphabet Σ, 2OTA over Σ define picture languages and compute the family of recognizable picture languages. Here, one could also define a transition function $\delta : Q \times Q \times \Sigma \to 2^Q$, instead of the set of transitions $T \subseteq Q \times Q \times \Sigma \times Q$. In case $|I| = 1$ and $\delta : Q \times Q \times \Sigma \to Q$, the automaton $\mathcal{A}$ is called a *deterministic* 2OTA over the alphabet Σ, and we say that the respective picture language is *deterministically* 2OTA *recognizable*.

The semiring-weighted version of this model was already investigated by Bozapalidis and Grammatikopoulou [24] and later by [52]. Here we consider picture valuation monoids as the weight structure and we we extend 2OTA to the weighted setting:

Definition 1.5. *A weighted 2-dimensional on-line tessellation automaton, for short* W2OTA*, $\mathcal{A} = (Q, T, I, F, \mathrm{wt})$ over an alphabet Σ and a pv-monoid $\mathbb{M}$ consists of a finite set Q of states, a set of transitions $T \subseteq Q \times Q \times \Sigma \times Q$, sets of initial and final states $I, F \subseteq Q$, respectively, and a weight function* $\mathrm{wt} : T \to M$.

For a transition $t = (q_h, q_v, a, q) \in T$, we set $\sigma_h(t) = q_h$, $\sigma_v(t) = q_v$ and $\sigma(t) = q$. Such a transition is depicted in Figure 1.1. We denote by $\mathrm{label}(t)$ its *label* a and by $\mathrm{wt}(t) = d$ its *weight* d. We extend both functions to pictures by setting for a picture $\rho = (c_{i,j})_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}} \in T^{m \times n}$ over the set of transitions:

$$\mathrm{label}(\rho)_{i,j} := \mathrm{label}(c_{i,j}) \text{ and } \mathrm{wt}(\rho) = \mathrm{val}(\mathrm{wt}(c_{i,j})_{i,j}).$$

This defines functions $\mathrm{label} : T^{++} \to \Sigma^{++}$ and $\mathrm{wt} : T^{++} \to M$. We call $\mathrm{label}(\rho)$ the *label* of ρ and $\mathrm{wt}(\rho)$ the *weight* of ρ. A *run* in $\mathcal{A}$ is an element in $T^{m \times n}$ satisfying

$$\sigma_h(c_{i,j}) = \sigma(c_{i,j-1}), \text{ for all } 1 \leqslant i \leqslant m,\ 2 \leqslant j \leqslant n,$$

$$\sigma_v(c_{i,j}) = \sigma(c_{i-1,j}), \text{ forall } 2 \leqslant i \leqslant m,\ 1 \leqslant j \leqslant n.$$

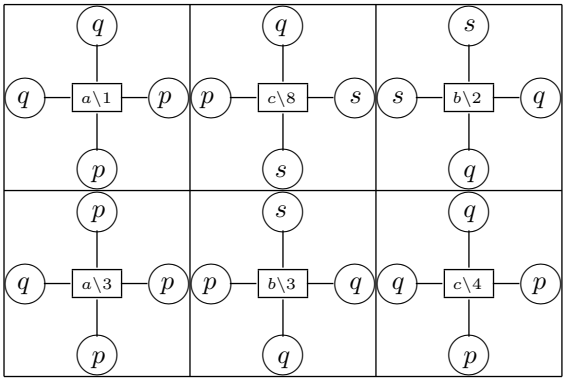


Figure 1.2: A run in an arbitrary W2OTA

We call the run $\rho \in T^{m\times n}$ *successful* if $\sigma_v(c_{1,j}), \sigma_h(c_{i,1}) \in I$ for all $1 \leqslant i \leqslant m$, $1 \leqslant j \leqslant n$ and $\sigma(c_{m,n}) \in F$. The set of all successful runs in $\mathcal{A}$ labeled with a picture p is denoted by $I \xrightarrow{p}_{\mathcal{A}} F$.

We define the *behavior* of $\mathcal{A}$ as the function $[\![\mathcal{A}]\!] : \Sigma^{++} \to M$ given by

$$[\![\mathcal{A}]\!](p) = \sum(\mathrm{wt}(\rho) \mid \rho \in I \xrightarrow{p}_{\mathcal{A}} F).$$

If $p \in \Sigma^{++}$ has no successful run in $\mathcal{A}$, then $[\![\mathcal{A}]\!](p) = \mathbf{0}$. Intuitively, the weight of a picture p is the sum of the weights of all successful runs in $\mathcal{A}$ that read p; note that the sum in the definition of the behaviour of $\mathcal{A}$ is the addition operation of the pv-monoid.

Example 1.6. *For instance, a run in an arbitrary* W2OTA *over the pv-monoid* $(\mathbb{R} \cup \{-\infty\}, \sup, \mathrm{avg}, -\infty)$ *and the alphabet* $\{a, b, c\}$ *where* p, q, s *are states, is depicted in Figure 1.2. The label of this run is* $\begin{pmatrix} a & c & b \\ a & b & c \end{pmatrix}$ *and its weight is* $\mathrm{avg}(\begin{pmatrix} 1 & 8 & 2 \\ 3 & 3 & 4 \end{pmatrix}) = 3.5$.

Definition 1.7. *Any function* $\mathbb{L} : \Sigma^{++} \to M$ *is called a picture series or a quantitative picture language over* Σ *and a pv-monoid* $\mathbb{M} = (M, +, \mathrm{val}, \mathbf{0})$. *If* $\mathbb{L} = [\![\mathcal{A}]\!]$ *for a* W2OTA $\mathcal{A}$ *taking weights in a subset* $E \subseteq M$, *then* $\mathbb{L}$ *is called* W2OTA*-recognizable over* E. *If* $\mathbb{L}$ *is* W2OTA*-recognizable over* M, *we call it* W2OTA*-recognizable.*

The concept of recognizability of a picture series over a subset of M is used in Section 1.5, where we deal with logic. In Section 1.5, we will define three different restrictions over conjunction and in order to deal with the third restriction we need to consider the concept of recognizability of a picture series over a subset of M.

Example 1.8. *Consider the pv-monoid* $(\mathbb{R} \cup \{-\infty\}, \sup, \text{avg}, -\infty)$ *with* $\text{avg}((d_{i,j})_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}}) = \frac{1}{m \times n} \sum_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}} d_{i,j}$. *Let* $\mathcal{A} = (\{q\}, T, \{q\}, \{q\}, \text{wt})$ *be a* W2OTA *over the alphabet* $\Sigma = \{b, w\}$ *with* $T = \{(q, q, b, q), (q, q, w, q)\}$, $\text{wt}(q, q, b, q) = -1$, *and* $\text{wt}(q, q, w, q) = 1$. *If we let the letter* b *interpret black color and the letter* w *interpret white color, then* $\mathcal{A}$ *computes the average difference of brightness and darkness for every monochrome picture* $p \in \Sigma^{++}$. *For example* $[\![\mathcal{A}]\!]\left(\begin{pmatrix} b & b & w \\ w & b & w \\ w & w & b \end{pmatrix}\right) = \frac{1}{9}$, $[\![\mathcal{A}]\!]\left(\begin{pmatrix} w & w & w \\ w & w & w \\ w & w & w \end{pmatrix}\right) = 1$, *and* $[\![\mathcal{A}]\!]\left(\begin{pmatrix} b & b \\ w & w \end{pmatrix}\right) = 0$.

Now we want to show that we can extend the notion of normalized automata for words to 2-dimensiona on-line tessellation automata. We will use this notion in Lemma 1.31. However, this extension could also be of independent interest.

Definition 1.9. *Let* $\mathcal{A} = (Q, T, I, F)$ *be a* 2OTA *over the alphabet* Σ. *We call* $\mathcal{A}$ *initial state normalized if* $I = \{q_0\}$ *and for all transitions* (q_h, q_v, a, q) *in* T, *where* $a \in \Sigma$ *and* $q_h, q_v, q \in Q$, *we have:* $q \neq q_0$. *We call* $\mathcal{A}$ *final state normalized if* $F = \{q_f\}$ *and for all transitions* (q_h, q_v, a, q) *in* T, *where* $a \in \Sigma$ *and* $q_h, q_v, q \in Q$, *we have* $q_h \neq q_f$ *and* $q_v \neq q_f$.

Proposition 1.10. *Let* $\mathcal{A} = (Q, T, I, F)$ *be a* 2OTA *over the alphabet* Σ. *Then there exists,*

1. *an initial state normalized* 2OTA $\mathcal{A}_i$ *over* Σ *such that* $L(\mathcal{A}_i) = L(\mathcal{A})$.
2. *a final state normalized* 2OTA $\mathcal{A}_f$ *over* Σ *such that* $L(\mathcal{A}_f) = L(\mathcal{A})$.

Proof. 1. Let $\mathcal{A} = (Q, T, I, F)$ be a 2OTA over the alphabet Σ recognizing the picture language $L(\mathcal{A})$. Now we build an initial state normalized 2OTA $\mathcal{A}_i = (Q', T', I', F')$ by letting:

- $Q' = Q \cup \{q_0\}$ and $q_0 \notin Q$, $I' = \{q_0\}$, $F' = F$,
- $T' = T \cup \{(q_h, q_0, a, q) \mid a \in \Sigma,\ \exists q_v \in I : (q_h, q_v, a, q) \in T\}$
  $\cup \{(q_0, q_v, a, q) \mid a \in \Sigma,\ \exists q_h \in I : (q_h, q_v, a, q) \in T\}$
  $\cup \{(q_0, q_0, a, q) \mid a \in \Sigma,\ \exists q_h \in I,\ \exists q_v \in I : (q_h, q_v, a, q) \in T\}$.

We mention that the set of transitions T' is defined in a different way from the word case, since in 2OTA we are dealing with two dimensions and based on the definition of a 2OTA there are some initial states located vertically and horizontally in the first row and first column, respectively. Let $p \in \Sigma^{m\times n}$, and let

$$\rho = (c_{i,j})_{\substack{1\leq i\leq m\\ 1\leq j\leq n}} = (q^{(h)}_{i,j}, q^{(v)}_{i,j}, a_{i,j}, q_{i,j})_{\substack{1\leq i\leq m\\ 1\leq j\leq n}}$$

be a successful run of $\mathcal{A}$ over p, i.e., for all $1 \leqslant i \leqslant m$, $1 \leqslant j \leqslant n$ we have $\sigma_v(c_{1,j}) = q^{(v)}_{1,j} \in I$, $\sigma_h(c_{i,1}) = q^{(h)}_{i,1} \in I$ and $\sigma(c_{m,n}) \in F$. Now if we replace the transition $c_{1,1} = (q^{(h)}_{1,1}, q^{(v)}_{1,1}, a_{1,1}, q_{1,1})$ by $c'_{1,1} = (q_0, q_0, a_{1,1}, q_{1,1})$, the transitions $c_{1,j} = (q^{(h)}_{1,j}, q^{(v)}_{1,j}, a_{1,j}, q_{1,j})$, $j \geqslant 2$, by $c'_{1,j} = (q^{(h)}_{1,j}, q_0, a_{1,j}, q_{1,j})$ and the transitions $c_{i,1} = (q^{(h)}_{i,1}, q^{(v)}_{i,1}, a_{i,1}, q_{i,1})$, $i \geqslant 2$, by $c'_{i,1} = (q_0, q^{(v)}_{i,1}, a_{i,1}, q_{i,1})$, we obtain the successful run $\rho' = (c'_{i,j})_{\substack{1\leq i\leq m\\ 1\leq j\leq n}}$ of $\mathcal{A}_i$ over p. Therefore, $L(\mathcal{A}_i) = L(\mathcal{A})$.

2. Let $\mathcal{A} = (Q, T, I, F)$ be a 2OTA recognizing the picture language $L(\mathcal{A})$. Now build a final state normalized 2OTA $\mathcal{A}_f = (Q', T', I', F')$ with:

- $Q' = Q \cup \{q'\}$ and $q' \notin Q$, $I' = I$, $F' = \{q'\}$,
- $T' = T \cup \{(q_h, q_v, a, q') \mid a \in \Sigma,\ \exists q \in F : (q_h, q_v, a, q) \in T\}$.

As in the previous case, it is not difficult to see that for a picture $p \in \Sigma^{m\times n}$ there is a successful run in $\mathcal{A}$ if and only if there is a successful run in $\mathcal{A}_f$. Therefore, $L(\mathcal{A}_f) = L(\mathcal{A})$. □

1.2 Closure Properties of Weighted Picture Automata

In this section, we consider several closure properties of recognizable picture series which we will use in the following three sections and which could be also of independent interest.

Let Σ and Γ be two sets. We can extend any mapping $\pi : \Sigma \to \Gamma$ to the function $\pi : \Sigma^{++} \to \Gamma^{++}$ in the natural way, mapping a picture $p \in \Sigma^{m\times n}$ to $p' \in \Gamma^{m\times n}$ such that $\pi(p(i,j)) = p'(i,j)$, for all $1 \leqslant i \leqslant m$, $1 \leqslant j \leqslant n$. Now let Σ and Γ be finite. We define for every $\mathbb{L} : \Sigma^{++} \to M$ the projection $\pi(\mathbb{L}) : \Gamma^{++} \to M$ by

$$\pi(\mathbb{L})(p) = \sum(\mathbb{L}(p') \mid p' \in \Sigma^{++}, \pi(p') = p)$$

for every $p \in \Gamma^{++}$. In addition, if $\mathbb{L}' : \Gamma^{++} \to M$, then we can define the inverse projection $\pi^{-1}(\mathbb{L}') : \Sigma^{++} \to M$ by $\pi^{-1}(\mathbb{L}')(p) = \mathbb{L}'(\pi(p))$, for every $p \in \Sigma^{++}$.

Proposition 1.11. *Let $\mathbb{M} = (M, +, \mathrm{val}, \mathbf{0})$ be a pv-monoid, and let $\pi : \Sigma \to \Gamma$ be a mapping.*

1. *If $\mathbb{L} : \Sigma^{++} \to M$ is* W2OTA*-recognizable over $E \subseteq M$, then so is $\pi(\mathbb{L}) : \Gamma^{++} \to M$.*
2. *If $\mathbb{L}' : \Gamma^{++} \to M$ is* W2OTA*-recognizable over $E \subseteq M$, then so is $\pi^{-1}(\mathbb{L}') : \Sigma^{++} \to M$.*

Proof. To prove part (1), we apply an idea used in [47]. Assume that $\mathcal{A}_{\mathbb{L}} = (Q_{\mathbb{L}}, T_{\mathbb{L}}, I_{\mathbb{L}}, F_{\mathbb{L}}, \mathrm{wt}_{\mathbb{L}})$ is a W2OTA over Σ such that $[\![A_{\mathbb{L}}]\!] = \mathbb{L}$. We construct a new W2OTA $\mathcal{A} = (Q, T, I, F, \mathrm{wt})$ over the alphabet Γ with

- $Q = Q_{\mathbb{L}} \times \Sigma$,
- $I = I_{\mathbb{L}} \times \{a_0^\star\}$ for some fixed $a_0^\star \in \Sigma$,
- $F = F_{\mathbb{L}} \times \Sigma$,
- $((q_h, a^\star), (q_v, b^\star), a, (q, c^\star)) \in T$ if and only if $\pi(c^\star) = a$ and $(q_h, q_v, c^\star, q) \in T_{\mathbb{L}}$,
- for every $((q_h, a^\star), (q_v, b^\star), a, (q, c^\star)) \in T$, we set:

$$\mathrm{wt}((q_h, a^\star), (q_v, b^\star), a, (q, c^\star)) = \mathrm{wt}_{\mathbb{L}}(q_h, q_v, c^\star, q).$$

Let $p \in \Gamma^{m \times n}$. We define the mapping

$$\Omega : (I \xrightarrow{p}_{\mathcal{A}} F) \quad \longrightarrow \bigcup_{p' \in \pi^{-1}(p)} (I_{\mathbb{L}} \xrightarrow{p'}_{\mathcal{A}_{\mathbb{L}}} F_{\mathbb{L}})$$

in the following way: Let

$$\rho = (c_{i,j})_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}} = ((q_{i,j}^{(h)}, a_{i,j}^\star), (q_{i,j}^{(v)}, b_{i,j}^\star), a_{i,j}, (q_{i,j}, c_{i,j}^\star))_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}}$$

be a successful run of $\mathcal{A}$ over $p \in \Gamma^{m \times n}$, i.e., over $p = (a_{i,j})_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}}$. Note that in the run ρ we have $a_{i,1}^\star = a_0^\star$, for $1 \leqslant i \leqslant m$, and $b_{1,j}^\star = a_0^\star$, for $1 \leqslant j \leqslant n$. By the construction of $\mathcal{A}$, we have $\pi((c_{i,j}^\star)_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}}) = (a_{i,j})_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}}$ and so,

$$\rho_{\mathbb{L}} = (c_{i,j}^{(\mathbb{L})})_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}} = (q_{i,j}^{(h)}, q_{i,j}^{(v)}, c_{i,j}^\star, q_{i,j})_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}}$$

is a successful run of $\mathcal{A}_{\mathbb{L}}$ with

$$\pi(\mathrm{label}(\rho_{\mathbb{L}})) = \pi(p') = \pi((c^{\star}_{i,j})_{\substack{1\leqslant i\leqslant m\\ 1\leqslant j\leqslant n}}) = (a_{i,j})_{\substack{1\leqslant i\leqslant m\\ 1\leqslant j\leqslant n}} = p.$$

It means $\mathrm{label}(\rho_{\mathbb{L}}) \in \pi^{-1}(p)$. So we put $\Omega(\rho) = \rho_{\mathbb{L}}$. It is easy to see that Ω is a bijection. In addition, from the definition of wt, we have,

$$\mathrm{wt}(\rho) = \mathrm{wt}_{\mathbb{L}}(\rho_{\mathbb{L}}) = \mathrm{wt}_{\mathbb{L}}(\Omega(\rho)),$$

for all $\rho \in I \xrightarrow{p}_{\mathcal{A}} F$. Now, we have:

$$\begin{aligned}
[\![\mathcal{A}]\!](p) &= \sum(\mathrm{wt}(\rho) \mid \rho \in I \xrightarrow{p}_{\mathcal{A}} F)\\
&= \sum(\mathrm{wt}(\Omega^{-1}(\rho_{\mathbb{L}})) \mid \rho_{\mathbb{L}} \in \bigcup_{p'\in\pi^{-1}(p)} (I_{\mathbb{L}} \xrightarrow{p'}_{\mathcal{A}_{\mathbb{L}}} F_{\mathbb{L}}))\\
&= \sum_{p'\in\pi^{-1}(p)} \sum(\mathrm{wt}_{\mathbb{L}}(\rho_{\mathbb{L}}) \mid \rho_{\mathbb{L}} \in I_{\mathbb{L}} \xrightarrow{p'}_{\mathcal{A}_{\mathbb{L}}} F_{\mathbb{L}})\\
&= \sum_{p'\in\pi^{-1}(p)} [\![\mathcal{A}_{\mathbb{L}}]\!](p') = \pi([\![\mathcal{A}_{\mathbb{L}}]\!])(p).
\end{aligned}$$

This shows that $\pi(\mathbb{L})$ is W2OTA-recognizable.

To prove part (2), let $\mathcal{A}_{\mathbb{L}'} = (Q_{\mathbb{L}'}, T_{\mathbb{L}'}, I_{\mathbb{L}'}, F_{\mathbb{L}'}, \mathrm{wt}_{\mathbb{L}'})$ be a W2OTA over Γ such that $[\![\mathcal{A}_{\mathbb{L}'}]\!] = \mathbb{L}'$. We construct a new W2OTA $\mathcal{A} = (Q_{\mathbb{L}'}, T, I_{\mathbb{L}'}, F_{\mathbb{L}'}, \mathrm{wt})$ over Σ such that

- $(q_h, q_v, a, q) \in T$ if and only if $(q_h, q_v, \pi(a), q) \in T_{\mathbb{L}'}$,
- $\mathrm{wt}(q_h, q_v, a, q) = \mathrm{wt}_{\mathbb{L}'}(q_h, q_v, \pi(a), q)$.

Now it can be easily seen that $[\![\mathcal{A}]\!] = \pi^{-1}([\![\mathcal{A}_{\mathbb{L}'}]\!])$. □

Definition 1.12. [7] *A weighted or unweighted* 2OTA $\mathcal{A}$ *is called unambiguous if for any picture there exists at most one successful run in* $\mathcal{A}$. *A picture language* L *is called unambiguously* 2OTA *recognizable if it can be recognized by an unambiguous* 2OTA. *A picture series is called unambiguously recognizable if it can be recognized by an unambiguous* W2OTA.

Let Σ be an alphabet, $\mathbb{M}$ a pv-monoid and $r : \Sigma \to M$ be a mapping. We denote by $\mathrm{val} \circ r : \Sigma^{++} \to M$ the picture series over $\mathbb{M}$ defined for all $p \in \Sigma^{++}$ by $(\mathrm{val} \circ r)(p) = \mathrm{val}(r(p))$.

Lemma 1.13. *Let* Σ *be an alphabet,* $\mathbb{M}$ *a pv-monoid and* $r : \Sigma \to M$ *a mapping. Then,* $\mathrm{val} \circ r$ *is unambiguously* W2OTA*-recognizable.*

Proof. Consider the W2OTA $\mathcal{A} = (Q, T, I, F, \mathrm{wt})$ over Σ and $\mathbb{M}$ such that:

- $Q = F = I = \{q\}$,
- $T = \{(q, q, a, q) \mid a \in \Sigma\}$,
- $\mathrm{wt}(q, q, a, q) = r(a)$, for some $a \in \Sigma$.

Let $p = (a_{i,j})_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}} \in \Sigma^{++}$. By the given construction, we have the successful run $\rho = (c_{i,j})_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}} = (q, q, a_{i,j}, q)_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}}$ of $\mathcal{A}$ over p. Clearly $\mathcal{A}$ is unambiguous and we have:

$$\begin{aligned}[\![\mathcal{A}]\!](p) &= \mathrm{wt}(\rho) = \mathrm{val}(\mathrm{wt}(c_{i,j})_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}}) \\ &= \mathrm{val}(r(a_{i,j})_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}}) = \mathrm{val}(r(p)) = (\mathrm{val} \circ r)(p)\end{aligned}$$

and therefore $[\![\mathcal{A}]\!] = \mathrm{val} \circ r$. Hence, $\mathrm{val} \circ r$ is unambiguously W2OTA-recognizable. □

Let $\mathbb{M} = (M, +, \mathrm{val}, \mathbf{0})$ be a pv-monoid and $\mathbb{L}, \mathbb{L}' : \Sigma^{++} \to M$. The *sum* $\mathbb{L} \oplus \mathbb{L}'$ is defined pointwise by $(\mathbb{L} \oplus \mathbb{L}')(p) = \mathbb{L}(p) + \mathbb{L}'(p)$ for all $p \in \Sigma^{++}$. Let $L \subseteq \Sigma^{++}$ be a picture language. Then the *intersection* $(\mathbb{L} \cap L) : \Sigma^{++} \to M$ is the picture series over $\mathbb{M}$ defined by $(\mathbb{L} \cap L)(p) = \mathbb{L}(p)$ if $p \in L$ and $(\mathbb{L} \cap L)(p) = \mathbf{0}$ if $p \in \Sigma^{++} \backslash L$.

Proposition 1.14. *Let $\mathbb{M} = (M, +, \mathrm{val}, \mathbf{0})$ be a pv-monoid. Let $\mathbb{L} : \Sigma^{++} \to M$ and $\mathbb{L}' : \Sigma^{++} \to M$ be* W2OTA*-recognizable picture series. Then $\mathbb{L} \oplus \mathbb{L}'$ is* W2OTA*-recognizable.*

Proof. Consider two W2OTA $\mathcal{A} = (Q, T, I, F, \mathrm{wt})$ and $\mathcal{A}' = (Q', T', I', F', \mathrm{wt}')$ recognizing $\mathbb{L}$ and $\mathbb{L}'$, respectively, i.e., $[\![\mathcal{A}]\!] = \mathbb{L}$ and $[\![\mathcal{A}']\!] = \mathbb{L}'$. We may assume $Q \cap Q' = \emptyset$. Then, clearly the disjoint union $\mathcal{A} \oplus \mathcal{A}' = (Q \cup Q', T \cup T', I \cup I', F \cup F', \mathrm{wt} \cup \mathrm{wt}')$ recognizes $\mathbb{L} \oplus \mathbb{L}'$. Therefore, $\mathbb{L} \oplus \mathbb{L}'$ is W2OTA-recognizable. □

Lemma 1.15. *Let $\mathbb{M} = (M, +, \mathrm{val}, \mathbf{0})$ be a pv-monoid, $\mathbb{L}$ a* W2OTA*-recognizable picture series and L a* 2OTA*-recognizable picture language.*

1. *If L is unambiguously* 2OTA*-recognizable, then the picture series $\mathbb{L} \cap L$ is* W2OTA*- recognizable.*
2. *If L is unambiguously* 2OTA*-recognizable and the picture series $\mathbb{L}$ is unambiguously* W2OTA*-recognizable, then $\mathbb{L} \cap L$ is unambiguously* W2OTA*-recognizable.*

3. If $\mathbb{M}$ is idempotent, then the series $\mathbb{L} \cap L$ is W2OTA*-recognizable.*

Proof. To prove this lemma, we use an extension of the product construction: Assume that $\mathcal{A} = (Q, T, I, F)$ is a 2OTA over Σ such that $L(\mathcal{A}) = L$, and $\mathcal{A}' = (Q', T', I', F', \mathrm{wt})$ is a W2OTA over Σ and $\mathbb{M}$ such that $[\![\mathcal{A}']\!] = \mathbb{L}$. Let $\tilde{\mathcal{A}} = (\tilde{Q}, \tilde{T}, \tilde{I}, \tilde{F}, \tilde{\mathrm{wt}})$ be the W2OTA over Σ and $\mathbb{M}$ where

- $\tilde{Q} = Q \times Q'$, $\tilde{I} = I \times I'$ and $\tilde{F} = F \times F'$,
- $((q_h, q'_h), (q_v, q'_v), a, (q, q')) \in \tilde{T}$ if and only if $(q_h, q_v, a, q) \in T$ and $(q'_h, q'_v, a, q') \in T'$,
- $\tilde{\mathrm{wt}}((q_h, q'_h), (q_v, q'_v), a, (q, q')) = \mathrm{wt}\,(q'_h, q'_v, a, q')$.

1. Let $p \in \Sigma^{++}$ and let $\mathcal{A}$ be unambiguous. Hence, $|I \xrightarrow{p}_{\mathcal{A}} F| \leqslant 1$, and so there is either no successful run on p in $\tilde{\mathcal{A}}$ or there are as many successful runs on $p \in \Sigma^{++}$ in $\tilde{\mathcal{A}}$ as there are in $\mathcal{A}'$. Then,

$$\begin{aligned}
[\![\tilde{\mathcal{A}}]\!](p) &= \sum(\tilde{\mathrm{wt}}(\rho) \mid \rho \in \tilde{I} \xrightarrow{p}_{\tilde{\mathcal{A}}} \tilde{F}) \\
&= \sum(\mathrm{wt}(\rho) \mid \rho \in I \xrightarrow{p}_{\mathcal{A}'} F) \\
&= [\![\mathcal{A}']\!](p) = \mathbb{L}(p) = (\mathbb{L} \cap L)(p).
\end{aligned}$$

2. Let $\mathcal{A}$ and $\mathcal{A}'$ be unambiguous. Then, it is clear that $\tilde{\mathcal{A}}$ is unambiguous. Now similar to (1), it can be seen that $[\![\tilde{\mathcal{A}}]\!](p) = (\mathbb{L} \cap L)(p)$, and therefore, $\mathbb{L} \cap L$ is unambiguously recognizable.

3. Let $\mathbb{M}$ be idempotent. Now consider a picture $p \in \Sigma^{++} \backslash L$. We get $I \xrightarrow{p}_{\mathcal{A}} F = \emptyset$, and hence $I \xrightarrow{p}_{\tilde{\mathcal{A}}} F = \emptyset$. Thus $[\![\tilde{\mathcal{A}}]\!](p) = (\mathbb{L} \cap L)(p) = \mathbf{0}$. If $p \in L$, then let $N = |I \xrightarrow{p}_{\mathcal{A}} F| > 0$. Then, every successful run on p in $\mathcal{A}'$ can be simulated by N runs in $\tilde{\mathcal{A}}$ having the same weights. If $\rho \in I \xrightarrow{p}_{\mathcal{A}'} F$, then by idempotency of $\mathbb{M}$ we have $\sum_{i=1}^{N} \mathrm{wt}(\rho) = \mathrm{wt}(\rho)$. Therefore,

$$\begin{aligned}
[\![\tilde{\mathcal{A}}]\!](p) &= \sum(\tilde{\mathrm{wt}}(\rho) \mid \rho \in \tilde{I} \xrightarrow{p}_{\tilde{\mathcal{A}}} \tilde{F}) \\
&= \sum \sum_{i=1}^{N} (\mathrm{wt}(\rho) \mid \rho \in I \xrightarrow{p}_{\mathcal{A}'} F) \\
&= \sum(\mathrm{wt}(\rho) \mid \rho \in I \xrightarrow{p}_{\mathcal{A}'} F) \\
&= [\![\mathcal{A}']\!](p) = \mathbb{L}(p) = (\mathbb{L} \cap L)(p).
\end{aligned}$$

□

Remark 1.16. *We note that the corresponding results with* W2OTA*-recognizability over a subset $E \subseteq M$ hold when $\mathbb{L}$ is a recognizable picture series over E.*

1.3 A Nivat Theorem for Weighted Picture Automata

In this section, we prove a Nivat-like theorem for recognizable picture series, i.e., for the behavior of W2OTA. Nivat's Theorem [83] (see also [15], Theorem 4.1) is one of the fundamental characterizations of rational transductions and establishes a connection between rational transductions and rational languages. Here, we establish a connection between picture series and recognizabe picture languages. Indeed, we will show that recognizable picture series can be obtained precisely as projections of particularly simple unambiguously recognizable series restricted to unambiguously recognizable picture languages. In addition, we show that if the underlying picture valuation monoid is idempotent, then we do not need unambiguity of the underlying picture language.

Let Σ be an alphabet and $\mathbb{M} = (M, +, \mathrm{val}, \mathbf{0})$ a pv-monoid. First we introduce the following abbreviations:

- $\mathbb{M}^{\mathrm{rec}}(\Sigma^{++}, \text{W2OTA})$ denotes the family of picture series recognized by W2OTA over Σ and $\mathbb{M}$.
- $\mathbb{M}^{\mathcal{N}}(\Sigma^{++})$ (with $\mathcal{N}$ meaning Nivat) stands for the set of all picture series $\mathbb{L} : \Sigma^{++} \to M$ over $\mathbb{M}$ such that there exist an alphabet Γ, mappings $\pi : \Gamma \to \Sigma$ and $r : \Gamma \to M$ and a recognizable picture language $L \subseteq \Gamma^{++}$ such that $\mathbb{L} = \pi((\mathrm{val} \circ r) \cap L)$.
- $\mathbb{M}^{\mathcal{N}}(\Sigma^{++}, \text{UNAMB})$ is defined like $\mathbb{M}^{\mathcal{N}}(\Sigma^{++})$ with the difference that L is an unambiguous picture language.

We prove the following inclusion between these families:

Lemma 1.17. *Let Σ be an alphabet and $\mathbb{M}$ a pv-monoid. Then*

$$\mathbb{M}^{\mathrm{rec}}(\Sigma^{++}, \text{W2OTA}) \subseteq \mathbb{M}^{\mathcal{N}}(\Sigma^{++}, \text{UNAMB}) \subseteq \mathbb{M}^{\mathcal{N}}(\Sigma^{++}).$$

Proof. Let $\mathcal{A} = (Q, T, I, F, \mathrm{wt})$ be a W2OTA over Σ and $\mathbb{M}$ such that $[\![\mathcal{A}]\!] = \mathbb{L}$. Let $\Gamma = T$. We define the mappings $\pi : \Gamma \to \Sigma$ and $r : \Gamma \to M$ for all $\lambda = (q_h, q_v, a, q) \in \Gamma$ by $\pi(\lambda) = a$ and $r(\lambda) = \mathrm{wt}(\lambda)$, so $r = \mathrm{wt}$. From $\mathcal{A}$ we construct the *2OTA* $\mathcal{A}' = (Q, T', I, F)$ over the enlarged alphabet Γ such that

$$T' = \{(q_h, q_v, (q_h, q_v, a, q), q) \mid (q_h, q_v, a, q) \in T,\ a \in \Sigma\}.$$

With this construction, clearly for every input label $(q_h, q_v, a, q) \in \Gamma$ there is at most one transition in T' with label (q_h, q_v, a, q). So $\mathcal{A}'$ is unambiguous and we put $L(\mathcal{A}') = L$. It remains to show that $\mathbb{L} = \pi((\mathrm{val} \circ r) \cap L)$. For this let $p \in \Sigma^{++}$. Note that $I \xrightarrow{p}_{\mathcal{A}} F = \{p' \in L(\mathcal{A}') \mid \pi(p') = p\}$. Moreover, if $\rho \in I \xrightarrow{p}_{\mathcal{A}} F$, we have $(\mathrm{val} \circ r)(\rho) = \mathrm{val}(r(\rho)) = \mathrm{wt}(\rho)$. Therefore,

$$\begin{aligned}\pi((\mathrm{val} \circ r) \cap L)(p) &= \sum((\mathrm{val} \circ r)(p') \mid p' \in L \text{ and } \pi(p') = p) \\ &= \sum(\mathrm{wt}(\rho) \mid \rho \in I \xrightarrow{p}_{\mathcal{A}} F) \\ &= [\![\mathcal{A}]\!](p).\end{aligned}$$

The final inclusion is trivial. □

Theorem 1.18. (Nivat theorem for picture series.) *Let Σ be an alphabet and $\mathbb{M}$ a pv-monoid. Then we have*

$$\mathbb{M}^{\mathrm{rec}}(\Sigma^{++}, \textsc{w2ota}) = \mathbb{M}^{\mathcal{N}}(\Sigma^{++}, \textsc{Unamb}) \subseteq \mathbb{M}^{\mathcal{N}}(\Sigma^{++}).$$

Moreover, if $\mathbb{M}$ is idempotent, then $\mathbb{M}^{\mathrm{rec}}(\Sigma^{++}, \textsc{w2ota}) = \mathbb{M}^{\mathcal{N}}(\Sigma^{++})$.

Proof. By Lemma 1.17, we have,

$$\mathbb{M}^{\mathrm{rec}}(\Sigma^{++}, \textsc{w2ota}) \subseteq \mathbb{M}^{\mathcal{N}}(\Sigma^{++}, \textsc{Unamb}) \subseteq \mathbb{M}^{\mathcal{N}}(\Sigma^{++}).$$

By applying Lemmas 1.13 and 1.15(1) and Proposition 1.11 we obtain $\mathbb{M}^{\mathcal{N}}(\Sigma^{++}, \textsc{Unamb}) \subseteq \mathbb{M}^{\mathrm{rec}}(\Sigma^{++}, \textsc{w2ota})$. Now assume $\mathbb{M}$ is idempotent. Then by Lemmas 1.13 and 1.15(3) and Proposition 1.11 we get $\mathbb{M}^{\mathcal{N}}(\Sigma^{++}) \subseteq \mathbb{M}^{\mathrm{rec}}(\Sigma^{++}, \textsc{w2ota})$. □

Let Σ be an alphabet and $\mathbb{M} = (M, +, \mathrm{val}, \mathbf{0})$ a pv-monoid. We denote by $\mathbb{M}^{\mathscr{P}}(\Sigma^{++}, \textsc{Unamb})$ (with $\mathscr{P}$ meaning projection) the family of picture series $\mathbb{L} : \Sigma^{++} \to M$ over $\mathbb{M}$ such that there exist an alphabet Γ, a mapping $\pi : \Gamma \to \Sigma$ and an unambiguously recognizable picture series $\mathbb{L}' : \Gamma^{++} \to M$ over $\mathbb{M}$ such that $\mathbb{L} = \pi(\mathbb{L}')$. Now as a corollary from Theorem 1.18, we show that the projections of unambiguously W2OTA-recognizable picture series are W2OTA-recognizable picture series, and vice versa.

Corollary 1.19. *Let Σ be an alphabet and $\mathbb{M} = (M, +, \mathrm{val}, \mathbf{0})$ a pv-monoid. Then $\mathbb{M}^{\mathscr{P}}(\Sigma^{++}, \textsc{Unamb}) = \mathbb{M}^{\mathrm{rec}}(\Sigma^{++}, \textsc{w2ota})$.*

Proof. The inclusion $\mathbb{M}^{\mathscr{P}}(\Sigma^{++}, \textsc{Unamb}) \subseteq \mathbb{M}^{\mathrm{rec}}(\Sigma^{++}, \textsc{w2ota})$ is implied by Proposition 1.11. To prove the other inclusion, assume that $\mathbb{L}$ is W2OTA-recognizable. Then by Nivat's theorem we have $\mathbb{L} \in \mathbb{M}^{\mathcal{N}}(\Sigma^{++}, \textsc{Unamb})$. So there exists an alphabet Γ, mappings $\pi : \Gamma \to \Sigma$

and $r : \Gamma \to D$ and an unambiguous picture language $L \subseteq \Gamma^{++}$ such that $\mathbb{L} = \pi((\mathrm{val} \circ r) \cap L)$. From Lemma 1.13, we know that $\mathrm{val} \circ r$ is unambiguously W2OTA-recognizable. We put $\mathbb{L}' = (\mathrm{val} \circ r) \cap L$. Then $\mathbb{L} = \pi(\mathbb{L}')$, and by Lemma 1.15(2), $\mathbb{L}'$ is unambiguously recognizable. Hence, $\mathbb{L} \in \mathbb{M}^{\mathscr{P}}(\Sigma^{++}, \mathrm{UNAMB})$. □

1.4 Weighted Existential MSO-Logic

In this section we introduce the syntax and semantics of the weighted MSO logic on pictures. The syntax is a combination of ideas introduced in [19], [41], [44], [45] and [52]. In [41, 52], disjunction and existential quantification were interpreted by the sum, and the semantics of both conjunction and universal quantification were defined by the product operation of the semiring. Here, we consider picture valuation monoids as the abstract model; hence, in the weighted MSO logic which we define the semantics of universal quantification will be interpreted by the picture valuation function, which for example provides the average value of light of a picture. We first need to equip our picture valuation monoid with a product operation and a unit element:

Definition 1.20. *A product picture valuation monoid, or ppv-monoid for short, is a tuple* $\mathbb{M} = (M, +, \mathrm{val}, \diamond, \mathbf{0}, \mathbf{1})$ *consisting of a pv-monoid* $(M, +, \mathrm{val}, \mathbf{0})$*, a binary operation* $\diamond : M^2 \to M$*, and* $\mathbf{1} \in M$ *with* $\mathrm{val}((\mathbf{1})_{i,j})_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}} = \mathbf{1}$ *for all* $m, n \geqslant 1$ *and* $\mathbf{0} \diamond d = d \diamond \mathbf{0} = \mathbf{0}$*,* $\mathbf{1} \diamond d = d \diamond \mathbf{1} = d$ *for all* $d \in M$*.*

Here we fix an alphabet Σ. For a picture p of size (m, n), the domain $\mathrm{dom}(p) = [m] \times [n]$ consists of all positions or pixels of p. For describing picture languages by logical formulas we use a a countable set $\mathcal{V}_i$ of *first-order variables* and a countable set $\mathcal{V}_s$ of *second-order variables*. First-order variables will always be interpreted as positions of a picture (i.e., with elements in $\mathrm{dom}(p)$), while second-order variables will be interpreted with sets of positions of a picture (i.e., with subsets of $\mathrm{dom}(p)$). We will use letters like $x, y, z, x_1, x_2, \ldots$ to denote first-order variables, and we will use letters like $X, Y, Z, X_1, X_2, \ldots$ to denote second-order variables.

Definition 1.21. *The syntax of weighted first-order logic over* Σ *and* $\mathbb{M}$ *is defined as follows:*

$$\beta ::= P_a(x) \mid xS_v y \mid xS_h y \mid x \in X \mid x = y \mid \neg\beta \mid \beta \wedge \beta \mid \forall x \beta$$
$$\varphi ::= d \mid \beta \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x \varphi \mid \forall x \varphi$$

where $d \in M$, $a \in \Sigma$, $x, y, X \in \mathcal{V}$. The formulas β are called boolean first-order formulas over Σ, for short $\mathrm{Bool}[\Sigma]$*, and the formulas φ are called weighted first-order formulas over Σ and $\mathbb{M}$, for short* $\mathrm{wFO}[\Sigma, \mathbb{M}]$*. Note that negation is only applied in boolean first-order formulas. Weighted existential monadic second-order logic over Σ and $\mathbb{M}$, for short* $\mathrm{wEMSO}[\Sigma, \mathbb{M}]$*, is defined to be the set of all formulas of the form $\exists X_1 \cdots \exists X_k \, \varphi$ where $k \geqslant 0$, $X_1, \cdots, X_k$ are second-order variables and $\varphi \in \mathrm{wFO}[\Sigma, \mathbb{M}]$.*

Given a formula $\psi \in \mathrm{wEMSO}[\Sigma, \mathbb{M}]$, the set *free*$(\psi)$ of all *free variables* of ψ is defined as usual. We say that ψ is a sentence if *free*$(\psi) = \emptyset$. Let us first recall the semantics of the first-order formulas in the classical setting. For a picture p of size (m, n), we denote the component of p at position $(i, j) \in \mathrm{dom}(p)$ by $p(i, j)$ or $p_{i,j}$. For every $a \in \Sigma$, the unary relation P_a consists of all positions $(i, j) \in \mathrm{dom}(p)$ with $p_{i,j} = a$ (i.e., a is the letter in row i and column j of p). S_v is the vertical successor relation on the positions of p, i.e., it consists of all tuples of positions in $\mathrm{dom}(p)$ of the form $\big((i, j), (i{+}1, j)\big)$. S_h is the horizontal successor relation on the positions of p, i.e., it consists of all tuples of positions in $\mathrm{dom}(p)$ of the form $\big((i, j), (i, j{+}1)\big)$. The intended meaning of the formulas $P_a(x)$, xS_vy, xS_hy, $x \in X$ and $x = y$ is respectively as follows:

- *the letter at position x is a,*
- *position y is the vertical successor of position x* (same column, next row),
- *position y is the horizontal successor of position x* (same row, next column),
- *position x belongs to the set X,*
- *x and y are interpreted by the same position.*

Formulas containing no set quantification but possibly including atomic formulas of the form $x \in X$ are called *first-order formulas*. First-order formulas are closed under existential and universal quantification of first-order variables, i.e., whenever φ is a first-order formula and $x \in \mathcal{V}_i$, then $\exists x \, \varphi$ and $\forall x \, \varphi$ are also first-order formulas. The intended meaning of these formulas is respectively as follows:

- *there exists a position x such that the statement made by φ is true,*
- *for all positions x, the statement made by φ is true.*

In the classical setting, the set EMSO of *existential monadic second-order formulas* consists of formulas ψ of the form $\exists X_1 \cdots \exists X_k \, \varphi$ where $k \geqslant 0$, $X_1, \cdots, X_k$ are second-order variables and φ contains only first-order quantifiers. The picture language $L(\varphi)$ defined by the sentence φ is the set of all pictures $p \in \Sigma^{++}$ satisfying φ. We call a picture language L *first-order*, for short FO (respectively EMSO)-*definable* if there is a first-order (respectively EMSO)-sentence φ such that $L = L(\varphi)$.

Let $\mathcal{V} = \mathcal{V}_i \cup \mathcal{V}_s$. We define a $(\mathcal{V}, p)$-*assignment* as a function:

$$\sigma : \mathcal{V} \to \mathrm{dom}(p) \cup 2^{\mathrm{dom}(p)}$$

which maps first-order variables in $\mathcal{V}$ to elements of $\mathrm{dom}(p)$ and second-order variables in $\mathcal{V}$ to subsets of $\mathrm{dom}(p)$. If $x \in \mathcal{V}_i$ and $(i,j) \in \mathrm{dom}(p)$, then the update $\sigma[x/(i,j)]$ for $(i,j) \in \mathbb{N} \times \mathbb{N}$ is defined as $\sigma[x/(i,j)](x) = (i,j)$ and $\sigma[x/(i,j)] \restriction_{\mathcal{V}\setminus\{x\}} = \sigma \restriction_{\mathcal{V}\setminus\{x\}}$. Similarly, the update $\sigma[X/I]$ is defined fo $I \subseteq \mathbb{N} \times \mathbb{N}$. We encode a pair (p, σ), where σ is a $(\mathcal{V}, p)$-assignment, as a picture over the enriched alphabet $\Sigma_{\mathcal{V}} = \Sigma \times \{0,1\}^{\mathcal{V}}$. Conversely, an element $r \in \Sigma_{\mathcal{V}}^{++}$ can be viewed as a pair (p, σ) where $p \in \Sigma^{++}$ is the projection over Σ and $\sigma \in (\{0,1\}^{\mathcal{V}})^{++}$ is the projection of r over $\{0,1\}^{\mathcal{V}}$. Now, if the latter projection $\sigma \in (\{0,1\}^{\mathcal{V}})^{++}$ is such that for each first-order variable $x \in \mathcal{V}$, the projection of σ to the x-coordinate contains exactly one pixel carrying a 1, then σ will be called a *valid* assignment.

Now, similar to [52] we give the semantics of wEMSO formulas. However, here the semantics of universal quantification will be interpreted by a valuation function.

Definition 1.22. *Let ψ be a formula in* $\mathrm{wEMSO}[\Sigma, \mathbb{M}]$ *and $\mathcal{V}$ be a finite set of variables containing free(ψ). The semantics of ψ will be a series $[\![\psi]\!]_{\mathcal{V}} : \Sigma_{\mathcal{V}}^{++} \to M$ such that if σ is not a valid $(p, \mathcal{V})$-assignment, then $[\![\psi]\!]_{\mathcal{V}}(p, \sigma) = \mathbf{0}$. Otherwise, we define $[\![\psi]\!]_{\mathcal{V}}(p, \sigma) \in M$ inductively as in Table 1.*

We write $\Sigma_\psi = \Sigma_{free(\psi)}$ and $[\![\psi]\!] = [\![\psi]\!]_{free(\psi)}$. In case ψ is a sentence, then the semantics is a picture series over Σ.

Let $\mathcal{D} = (D, +, val, \diamond, 0, 1)$ be a ppv-monoid and $S, S' : \Sigma^{++} \to D$. Then $S \diamond S'$ is defined pointwise by $(S \diamond S')(p) = S(p) \diamond S'(p)$ for all $p \in \Sigma^{++}$.

$$[\![d]\!]_{\mathcal{V}}(p,\sigma) = d$$

$$[\![P_a(x)]\!]_{\mathcal{V}}(p,\sigma) = \begin{cases} 1, & \text{if } p(\sigma(x)) = a, \\ 0, & \text{otherwise} \end{cases} \qquad [\![\neg\beta]\!]_{\mathcal{V}}(p,\sigma) = \begin{cases} 1, & \text{if } [\![\beta]\!]_{\mathcal{V}}(p,\sigma) = 0, \\ 0, & \text{if } [\![\beta]\!]_{\mathcal{V}}(p,\sigma) = 1 \end{cases}$$

$$[\![xS_v y]\!]_{\mathcal{V}}(p,\sigma) = \begin{cases} 1, & \text{if } \sigma(x) S_v \sigma(y), \\ 0, & \text{otherwise} \end{cases} \qquad [\![\varphi_1 \vee \varphi_2]\!]_{\mathcal{V}}(p,\sigma) = [\![\varphi_1]\!]_{\mathcal{V}}(p,\sigma) + [\![\varphi_2]\!]_{\mathcal{V}}(p,\sigma)$$

$$[\![xS_h y]\!]_{\mathcal{V}}(p,\sigma) = \begin{cases} 1, & \text{if } \sigma(x) S_h \sigma(y), \\ 0, & \text{otherwise} \end{cases} \qquad [\![\varphi_1 \wedge \varphi_2]\!]_{\mathcal{V}}(p,\sigma) = [\![\varphi_1]\!]_{\mathcal{V}}(p,\sigma) \diamond [\![\varphi_2]\!]_{\mathcal{V}}(p,\sigma)$$

$$[\![x = y]\!]_{\mathcal{V}}(p,\sigma) = \begin{cases} 1, & \text{if } \sigma(x) = \sigma(y), \\ 0, & \text{otherwise} \end{cases} \qquad [\![\exists x \varphi]\!]_{\mathcal{V}}(p,\sigma) = \sum_{(i,j) \in \mathrm{Dom}(p)} [\![\varphi]\!]_{\mathcal{V} \cup \{x\}}(p, \sigma[x/(i,j)])$$

$$[\![\forall x \varphi]\!]_{\mathcal{V}}(p,\sigma) = val\Big([\![\varphi]\!]_{\mathcal{V} \cup \{x\}}(p, \sigma[x/(i,j)])_{(i,j) \in \mathrm{Dom}(p)}\Big)$$

$$[\![x \in X]\!]_{\mathcal{V}}(p,\sigma) = \begin{cases} 1, & \text{if } \sigma(x) \in \sigma(X), \\ 0, & \text{otherwise} \end{cases} \qquad [\![\exists X \varphi]\!]_{\mathcal{V}}(p,\sigma) = \sum_{I \subseteq \mathrm{Dom}(p)} [\![\varphi]\!]_{\mathcal{V} \cup \{X\}}(p, \sigma[X/I])$$

Table 1.1: The semantics of formulas in wEMSO$[\Sigma, \mathbb{M}]$

Example 1.23. *Consider the ppv-monoid* $(\mathbb{R} \cup \{-\infty\}, \sup, \mathrm{avg}, +, -\infty, 0)$ *with* $\mathrm{avg}((d_{i,j})_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}}) = \frac{1}{m \times n} \sum_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}} d_{i,j}$. *Let* $B \subseteq [0,1]$ *be a finite set of values as considered in Example 3.1. Let* B^{++} *be the set of all pictures over* B. *Consider the formula* $\psi = \forall x \, (\bigvee_{b \in B} (P_b(x) \wedge b))$. *Since universal quantification is interpreted by average, the semantics of the formula* ψ *provides the average value* $[\![\psi]\!](p)$ *of light of* p. *Therefore, we have* $[\![\psi]\!] = \mathbb{L}$ *for the picture series* $\mathbb{L}$ *from Example 3.1 (where* $L = B^{++}$*).*

1.5 Definability Equals Recognizability

We already know that for semirings and words (correspondingly also for pictures), the full wEMSO logic is expressively stronger than weighted automata [41]. Here, we also need to define suitable fragments of our wEMSO logic to obtain their equivalence to the family of recognizable picture series. We will define four different fragments: In all cases the application of universal first order quantification is restricted to the class of *almost boolean first-order formulas*, the smallest class containing all constant $d \in M$ and all formulas in Bool$[\Sigma]$ and which is closed under disjunction and conjunction. In one case we also restrict the use of constants in the formula by allowing their occurrence only in the scope of the first-order universal quantifier. The other fragments differ in the application of conjunction. Considering these restrictions makes our results different from the ones in [52]. In [52] the author considers commutative semirings. We show that our weighted automata device and

these fragments of wEMSO logic are expressively equivalent, under suitable assumptions on the underlying picture valuation monoid.

Definition 1.24. *Let Σ be an alphabet. For a language $L \subseteq \Sigma^{++}$, the characteristic series $\mathbf{1}_L : \Sigma^{++} \to M$ is defined for $p \in \Sigma^{++}$ as follows:*

$$\mathbf{1}_L(p) = \begin{cases} \mathbf{1} & \text{if } p \in L, \\ \mathbf{0} & \text{otherwise.} \end{cases}$$

Lemma 1.25. *Let $\mathbb{M} = (M, +, \mathrm{val}, \diamond, \mathbf{0}, \mathbf{1})$ be a ppv-monoid and L a* 2OTA*-recognizable picture language. If L is unambiguously* 2OTA*-recognizable, then $\mathbf{1}_L$ is unambiguously* W2OTA*-recognizable.*

Proof. Let $\mathcal{A} = (Q, T, I, F)$ be an unambiguous 2OTA-recognizing the picture language $L \subseteq \Sigma^{++}$. Then we denote $\mathcal{A}^{\mathbf{1}} = (Q, T, I, F, \mathrm{wt})$ the W2OTA where $\mathcal{A}$ is the underlying 2OTA assigning $\mathbf{1}$ to every transition in $\mathcal{A}$. Since $\mathcal{A}$ is unambiguous, for every $p \in \Sigma^{++}$ there is at most one successful run in $\mathcal{A}$. Therefore, we have:

$$[\![\mathcal{A}^{\mathbf{1}}]\!](p) = \begin{cases} \mathrm{val}((\mathbf{1})_{i,j})_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}} = \mathbf{1} & \text{if } p \in L, \\ \mathbf{0}, & \text{otherwise.} \end{cases}$$

So $[\![\mathcal{A}^{\mathbf{1}}]\!] = \mathbf{1}_L$. □

Remark 1.26. *Note that for any picture series $\mathbb{L} : \Sigma^{++} \to M$ and a picture language $L \subseteq \Sigma^{++}$, by Definition 1.24, we have $\mathbb{L} \diamond \mathbf{1}_L = \mathbb{L} \cap L$.*

Proposition 1.27. *Let ψ be a formula in* wEMSO$[\Sigma, \mathbb{M}]$ *and $\mathcal{V}$ a finite set of variables with free$(\psi) \subseteq \mathcal{V}$. If σ is a valid $(p, \mathcal{V})$-assignment, then we have $[\![\psi]\!]_{\mathcal{V}}(p, \sigma) = [\![\psi]\!](p, \sigma \restriction_{free(\psi)})$. In addition, the series $[\![\psi]\!]$ is* W2OTA*-recognizable over $E \subseteq M$ if and only if $[\![\psi]\!]_{\mathcal{V}}$ is* W2OTA*-recognizable over $E \subseteq M$.*

Proof. The first assertion can be shown by induction on ψ, as in [52] for semirings. For the second claim, let $[\![\psi]\!]$ be W2OTA-recognizable over E. Consider the homomorphism $\pi : \Sigma_{\mathcal{V}}^{++} \to \Sigma_{\psi}^{++}$ which erases components of $\mathcal{V} \backslash free(\psi)$ in a letter of $\Sigma_{\mathcal{V}}$. Let

$$N_{\mathcal{V}} = \{(p, \sigma) \in \Sigma_{\mathcal{V}}^{++} \mid \sigma \text{ is a valid assignment}\}.$$

The language $N_{\mathcal{V}}$ is recognizable by a deterministic picture automaton where the rules verify whether the input picture in $\Sigma_{\mathcal{V}}^{++}$ contains precisely

one 1 in every layer belonging to an first-order variable in $\mathcal{V}$. Then we have $[\![\psi]\!]_{\mathcal{V}} = \pi^{-1}([\![\psi]\!]) \cap N_{\mathcal{V}}$ which means $[\![\psi]\!]_{\mathcal{V}}$ is W2OTA-recognizable over E, because of our first claim and due to Proposition 1.11(2) and Lemma 1.15(1).

Conversely, let $[\![\psi]\!]_{\mathcal{V}}$ be W2OTA-recognizable over E. Let F be the set of all pictures $(p, \sigma) \in N_{\mathcal{V}}$ such that σ assigns to each variable x (resp. X) in $\mathcal{V} \backslash \mathit{free}(\psi)$ coordinate $(1,1)$, i.e., $\sigma(x) = (1,1)$ (resp. $\sigma(X) = \{(1,1)\}$). Then F is deterministically recognizable, and we have $[\![\psi]\!] = \pi([\![\psi]\!]_{\mathcal{V}} \cap F)$. By Lemma 1.25, Lemma 1.15(1) and Proposition 1.11(1), $[\![\psi]\!]$ is W2OTA-recognizable over E. □

Definition 1.28. *A picture series $\mathbb{L} : \Sigma^{++} \to M$ is a first-order step function (*FO *step function) if $\mathbb{L} = \bigoplus_{i=1}^{n} d_i \cdot \mathbf{1}_{L_i}$ for some $n \in \mathbb{N}$, $d_i \in M$ and* FO*-definable picture languages $L_i \subseteq \Sigma^{++} (i = 1, ..., n)$ forming a partition of Σ^{++}. We call $\mathbb{L}$ a boolean step function if $\mathbb{L}$ is an* FO *step function with $\mathbb{L}(\Sigma^{++}) \subseteq \{\mathbf{0}, \mathbf{1}\}$.*

Let $\beta \in \mathrm{Bool}[\Sigma]$. We may regard β as a classical boolean first-order formula defining the picture language $L(\beta)$. Let $(p, \sigma) \in \Sigma_{\beta}^{++}$. If $(p, \sigma) \notin N_{\beta}$ then $[\![\beta]\!](p, \sigma) = \mathbf{0}$ and $(p, \sigma) \notin L(\beta)$. Now assume $(p, \sigma) \in N_{\beta}$. By structural induction on β one can easily see that $[\![\beta]\!](p, \sigma) = \mathbf{1}$ if $(p, \sigma) \models \beta$, and equals $\mathbf{0}$ otherwise. This shows $[\![\beta]\!] = \mathbf{1}_{L(\beta)}$ which is a boolean step function.

Now, as in [44], we consider underlying ppv-monoids with some richer conditions. We use these conditions in order to get more closure properties of recognizable picture series which we will use in the proof of our main result of this section.

A ppv-monoid $\mathbb{M} = (M, +, \mathrm{val}, \diamond, \mathbf{0}, \mathbf{1})$ is *left-+-distributive* if

$$d \diamond (d_1 + d_2) = (d \diamond d_1) + (d \diamond d_2)$$

for any $d, d_1, d_2 \in M$; right-+-distributivity is defined analogously. If $\mathbb{M}$ is both left- and right-+-distributive, then $\mathbb{M}$ is *+-distributive*. If $\diamond$ is associative, then $\mathbb{M}$ is called *associative*. Moreover, $\mathbb{M}$ is *left-multiplicative* if for all $m,\ n \geqslant 1$ and $d, d_{i,j} \in M$

$$d \diamond \mathrm{val}(d_{1,1}, d_{1,2}, ..., d_{1,n}, d_{2,1}, ..., d_{2,n}, ..., d_{m,1}, ..., d_{m,n}) =$$

$$\mathrm{val}(d \diamond d_{1,1}, d_{1,2}, ..., d_{1,n}, d_{2,1}, ..., d_{2,n}, ..., d_{m,1}, ..., d_{m,n}).$$

We call $\mathbb{M}$ *left-val-distributive* if for all $m,\ n \geqslant 1$ and $d, d_{i,j} \in M$

$$d \diamond \mathrm{val}(d_{1,1}, ..., d_{1,n}, d_{2,1}, ..., d_{2,n}, ..., d_{m,1}, ..., d_{m,n}) =$$
$$\mathrm{val}(d \diamond d_{1,1}, ..., d \diamond d_{1,n}, d \diamond d_{2,1}, ..., d \diamond d_{2,n}, ..., d \diamond d_{m,1}, ..., d \diamond d_{m,n}).$$

If $d \diamond d' = d' \diamond d$, we say that d and d' *commute*. Let $C, C' \subseteq M$. If $d \diamond d' = d' \diamond d$ for all $d \in C$ and $d' \in C'$, then C and C' *commute*. We call a ppv-monoid $\mathbb{M}$ *conditionally commutative* if for any $m,\ n \geqslant 1$ and any two sequences $(d_{1,1}, ..., d_{1,n}, d_{2,1}, ..., d_{2,n}, ..., d_{m,1}, ..., d_{m,n})$ and $(d'_{1,1}, ..., d'_{1,n}, d'_{2,1}, ..., d'_{2,n}, ..., d'_{m,1}, ..., d'_{m,n})$ from M with $d_{i,j} \diamond d'_{k,l} = d'_{k,l} \diamond d_{i,j}$ for all $1 \leqslant i, k \leqslant m$ and $1 \leqslant j, l \leqslant n$, we have

$$\mathrm{val}(d_{1,1}, ..., d_{1,n}, ..., d_{m,1}, ..., d_{m,n}) \diamond \mathrm{val}(d'_{1,1}, ..., d'_{1,n}, ..., d'_{m,1}, ..., d'_{m,n})$$
$$= \mathrm{val}(d_{1,1} \diamond d'_{1,1}, ..., d_{1,n} \diamond d'_{1,n}, ..., d_{m,1} \diamond d'_{m,1}, ..., d_{m,n} \diamond d'_{m,n}).$$

A ppv-monoid $\mathbb{M}$ is *left-distributive* if it is left-$+$-distributive, and moreover, left-multiplicative or left-val-distributive. Whenever $\mathbb{M}$ is $+$-distributive and associative, then $(M, +, \diamond, \mathbf{0}, \mathbf{1})$ is a semiring and we call $\mathbb{M} = (M, +, \mathrm{val}, \diamond, \mathbf{0}, \mathbf{1})$ a *picture valuation semiring*. A picture valuation semiring which is also left-multiplicative or left-val-distributive, and conditionally commutative is called a *conditionally commutative picture valuation semiring*, or for short cc-picture valuation semiring.

Let $\mathbb{M} = (M, +, \mathrm{val}, \diamond, \mathbf{0}, \mathbf{1})$ be a ppv-monoid. For $d \in M$ the picture series $d\mathbf{1}_{\Sigma^{++}}$ is defined by $d\mathbf{1}_{\Sigma^{++}}(p) = d$ for every $p \in \Sigma^{++}$. We call $d \in M$ *regular* if for any alphabet Σ there is a W2OTA $\mathcal{A}_d = (Q, T, I, F, \mathrm{wt})$ such that $[\![\mathcal{A}_d]\!] = d1_{\Sigma^{++}}$ and $\mathrm{wt}(T) \subseteq \{\mathbf{1}, d\}$. $\mathbb{M}$ is *regular* if every $d \in M$ is regular.

For example, if $(M, +, \cdot, \mathbf{0}, \mathbf{1})$ is a semiring and $\mathrm{val} : M^{++} \to M$ the natural product, then $(M, +, \mathrm{val}, \cdot, \mathbf{0}, \mathbf{1})$ is a left-multiplicative picture valuation semiring. As another example, $(\mathbb{R} \cup \{-\infty\}, \sup, \mathrm{avg}, +, -\infty, 0)$ is a cc-picture valuation semiring. Now consider the pv-monoid $(\mathbb{N} \cup \{\infty\}, \min, \mathrm{maj}, \infty)$ in Example 3.2. If we add $\mathbf{1} = -\infty$ and we define max as the product operation, then we get a ppv-monoid which is regular, $+$-distributive, associative but neither left-multiplicative nor left-val-distributive nor conditionally commutative. For many further examples, we refer the reader to [44].

We note that if $\mathbb{M}$ is left-val-distributive or left-multiplicative, then $\mathbb{M}$ is regular: Indeed, in case $\mathbb{M}$ is left-multiplicative, we construct the automaton

$\mathcal{A} = (\{q_0, q_1\}, T, \{q_0\}, \{q_1\}, \mathrm{wt})$ such that

$$T = \bigcup_{a \in \Sigma} \{(q_0, q_0, a, q_1), (q_0, q_1, a, q_1), (q_1, q_0, a, q_1), (q_1, q_1, a, q_1)\}$$

and for $t \in T$ we have:

$$\mathrm{wt}(t) = \begin{cases} d & \text{if } t = (q_0, q_0, a, q_1), \\ \mathbf{1} & \text{otherwise.} \end{cases}$$

Clearly, the W2OTA $\mathcal{A}$ computes $d \cdot \mathbf{1}_{\Sigma^{++}}$, i.e., $[\![\mathcal{A}]\!] = d \cdot \mathbf{1}_{\Sigma^{++}}$. Hence, $\mathbb{M}$ is regular. In the case that $\mathbb{M}$ is left-val-distributive, we consider the same construction, but this time to all transitions in T we assign weight d. Then it is clear that the W2OTA $\mathcal{A}$ computes $d \cdot \mathbf{1}_{\Sigma^{++}}$. Therefore, $\mathbb{M}$ is regular.

Proposition 1.29. [52] *Let Σ be any alphabet and let φ be a first-order formula. Then $\mathcal{L}(\varphi)$ is an unambiguously recognizable picture language.*

Lemma 1.30. *Let $\mathbb{M} = (M, +, \mathrm{val}, \diamond, \mathbf{0}, \mathbf{1})$ be a ppv-monoid. Then the class of* FO *step functions is closed under $+$ and $\diamond$. If $\mathbb{M}$ is regular, then every* FO *step function $\mathbb{L}$ over $\mathbb{M}$ is a* W2OTA*-recognizable picture series.*

Proof. It can be proved by the same approach as Lemma 2.12 in [44], using the fact that FO-definable picture languages are closed under intersection, and also applying Proposition 1.29, Lemma 1.15(1) and Proposition 1.14. □

We define the product automata construction which we use in the following lemmas. Let $\mathcal{A}_1 = (Q_1, T_1, I_1, F_1, \mathrm{wt}_1)$ and $\mathcal{A}_2 = (Q_2, T_2, I_2, F_2, \mathrm{wt}_2)$ be two W2OTA over the alphabet Σ and ppv-momoid $\mathbb{M}$. Then the product automaton of $\mathcal{A}_1$ and $\mathcal{A}_2$ is the W2OTA $\mathcal{A} = (Q_1 \times Q_2, T, I_1 \times I_2, F_1 \times F_2, \mathrm{wt})$ where

- $((q_h, q'_h), (q_v, q'_v), a, (q, q')) \in T$ if and only if $(q_h, q_v, a, q) \in T_1$ and $(q'_h, q'_v, a, q') \in T_2$,
- $\mathrm{wt}((q_h, q'_h), (q_v, q'_v), a, (q, q')) = \mathrm{wt}_1(q_h, q_v, a, q) \diamond \mathrm{wt}_2(q'_h, q'_v, a, q')$.

We also need to define $E \diamond E' = \{e \diamond e' | e \in E, e' \in E'\}$ for $E, E' \subseteq M$.

Lemma 1.31. *Let $\mathbb{M}$ be a left distributive ppv-monoid, $E \subseteq M$, $\mathbb{L}$ an* FO *step function and $\mathbb{L}'$* W2OTA*-recognizable over E'. Then $\mathbb{L} \diamond \mathbb{L}'$ is* W2OTA*-recognizable over* $\mathrm{im}(\mathbb{L}) \diamond E'$.

Proof. Let $\mathbb{L} = \bigoplus_{i=1}^{n} d_i \mathbf{1}_{L_i}$ be an FO step function. Then, we have:

$$\begin{aligned}\mathbb{L} \diamond \mathbb{L}'(p) &= (\bigoplus_{i=1}^{n} d_i \mathbf{1}_{L_i})(p) \diamond \mathbb{L}'(p) \\ &= \bigoplus_{i=1}^{n} (d_i \mathbf{1}_{L_i})(p) \diamond \mathbb{L}'(p) \\ &= \begin{cases} d_1 \diamond \mathbb{L}'(p) & \text{if } p \in L_1, \\ \vdots & \\ d_n \diamond \mathbb{L}'(p) & \text{if } p \in L_n. \end{cases}\end{aligned}$$

By using left-distributivity of $\mathbb{M}$ and with the product construction given above, we will show that $(d\mathbf{1}_L) \diamond \mathbb{L}'$ is W2OTA-recognizable picture series over $d \diamond E'$ for any $d \in M$ and any unambiguously recognizable picture language L. Then the closure of recognizable picture series under sum, cf. Proposition 1.14, will yield the result. First, we let $\mathbb{M}$ be a left-multiplicative ppv-monoid. We construct the W2OTA $\mathcal{A}_L = (Q_L, T_L, \{q_L\}, F_L, \mathrm{wt}_L)$ where, by Proposition 1.10, $(Q_L, T_L, \{q_L\}, F_L)$ is an initial state normalized unambiguous 2OTA recognizing the FO-definable language L, and

$$\mathrm{wt}_L(q_h, q_v, a, q) = \begin{cases} d & \text{if } q_h = q_v = q_L \neq q, \\ \mathbf{1} & \text{otherwise.} \end{cases}$$

Now we let $\mathcal{A}_{\mathbb{L}'} = (Q_{\mathbb{L}'}, T_{\mathbb{L}'}, I_{\mathbb{L}'}, F_{\mathbb{L}'}, \mathrm{wt}_{\mathbb{L}'})$ be a W2OTA recognizing $\mathbb{L}'$. We can construct the product automaton $\mathcal{A} = (Q, T, I, F, \mathrm{wt})$ of $\mathcal{A}_L$ and $\mathcal{A}_{\mathbb{L}'}$ with the above construction. Now by the same approach as in Lemma 2.13 of [44], we obtain that $[\![\mathcal{A}]\!] = [\![\mathcal{A}_L]\!] \diamond [\![\mathcal{A}_{\mathbb{L}}]\!] = \mathbb{L} \diamond \mathbb{L}'$.

The case that $\mathbb{M}$ is left-val-distributive can be handled by the same approach. □

Lemma 1.32. *Let $\mathbb{M} = (M, +, \mathrm{val}, \diamond, \mathbf{0}, \mathbf{1})$ be a cc-picture valuation semiring, $E, E' \subseteq M$, $\mathbb{L}$ a* W2OTA*-recognizable picture series over E, and $\mathbb{L}'$ a* W2OTA*-recognizable picture series over E'. If E and E' commute, then $\mathbb{L} \diamond \mathbb{L}'$ is* W2OTA*-recognizable over $E \diamond E'$.*

Proof. This can be shown using the product construction and in a similar way to Lemma 2.14 of [44]. □

Lemma 1.33. *Let ψ be a formula in* WEMSO$[\Sigma, \mathbb{M}]$ *and $\mathcal{V}$ a finite set of variables with free$(\psi) \subseteq \mathcal{V}$. Then $[\![\psi]\!]$ is an* FO *step function if and only if $[\![\psi]\!]_{\mathcal{V}}$ is an* FO *step function.*

Proof. In [52], this result has been proved over commutative semirings. In the process of the proof there is no use of distributivity or associativity, so the same argument holds in our setting over picture valuation monoids. □

Now we want to define a class of formulas describing all FO step functions.

Definition 1.34. *The class of almost boolean first-order formulas of* wEMSO$[\Sigma, \mathbb{M}]$ *is the smallest class containing all constant* $d \in M$ *and all formulas in* Bool$[\Sigma]$ *and which is closed under disjunction and conjunction. We denote the set of all almost boolean formulas over* Σ *and* $\mathbb{M}$ *by* aBool$[\Sigma, \mathbb{M}]$

Proposition 1.35. *Let* φ *be a formula in* aBool$[\Sigma, \mathbb{M}]$. *Then* $[\![\varphi]\!]$ *is an* FO *step function. Conversely, let* $\mathbb{M}$ *be a ppv-monoid. If* $\mathbb{L} : \Sigma^{++} \to M$ *is an* FO *step function, then* $\mathbb{L} = [\![\varphi]\!]$ *for some sentence* $\varphi \in$ aBool$[\Sigma, \mathbb{M}]$.

Proof. We prove the first part by induction on the structure of φ. For $\varphi = d \in M$ we have $[\![\varphi]\!] = d1_{\Sigma^{++}}$ which is an FO step function. If φ is a boolean first-order formula, then it can be regarded as a classical boolean first-order formula defining the picture language $L(\varphi)$. Then $[\![\varphi]\!] = \mathbf{1}_{L(\varphi)}$. Now consider $\varphi \vee \psi$ and $\varphi \wedge \psi$ with $\mathcal{V} = \mathit{free}(\varphi \vee \psi) = \mathit{free}(\varphi \wedge \psi)$. If $[\![\varphi]\!]$ and $[\![\psi]\!]$ are FO step functions, then by Lemma 1.33, $[\![\varphi]\!]_{\mathcal{V}}$ and $[\![\psi]\!]_{\mathcal{V}}$ are FO step functions. Now due to Lemma 1.30, we obtain that $[\![\varphi \vee \psi]\!] = [\![\varphi]\!]_{\mathcal{V}} + [\![\psi]\!]_{\mathcal{V}}$ and $[\![\varphi \wedge \psi]\!] = [\![\varphi]\!]_{\mathcal{V}} \diamond [\![\psi]\!]_{\mathcal{V}}$ are FO step functions.

Conversely, let $\mathbb{L} = \bigoplus_{i=1}^{n} d_i 1_{L_i}$ be an FO step function. Since L_i $(1 \leqslant i \leqslant n)$ is FO-definable, there is a classical unweighted FO-sentence α_i such that $L(\alpha_i) = L_i$. If we replace existential quantification and disjunction in α_i by universal quantification, conjunction and negation, we obtain a boolean first-order sentence β_i with $[\![\beta_i]\!] = \mathbf{1}_{L_i}$. Since $\{L_1, ..., L_n\}$ is a partition, for the formula $\varphi = \bigvee_{i=1}^{n} (d_i \wedge \beta_i)$ we get $[\![\varphi]\!] = \mathbb{L}$. □

Let Const(φ) denote the set of elements from M occurring as subformulas in φ. Inspired by the ideas applied in [44, 45], we define the restriction form of wEMSO$[\Sigma, \mathbb{M}]$ as follows:

Definition 1.36. *A formula* ψ *in* wEMSO$[\Sigma, \mathbb{M}]$ *is* $\forall$*-restricted, if whenever it contains a subformula of the form* $\forall x\, \varphi$, *then* $\varphi \in$ aBool$[\Sigma, \mathbb{M}]$. *The formula* ψ *is called* M*-restricted if whenever it contains a subformula of the form* $d \in M$, *then there exists a subformula of* ψ *of the form* $\psi' = \forall x \varphi$ *such that* d *is a subformula of* φ, *i.e., every constant* $d \in M$ *occurring in* ψ *is in the scope of the first-order universal quantifier. In addition,* ψ *is called*

- *strongly $\wedge$-restricted, if for every subformula $\varphi_1 \wedge \varphi_2$ of ψ:*
  - *both φ_1 and φ_2 are formulas in* $\mathrm{aBool}[\Sigma, \mathbb{M}]$ *, or*
  - *φ_1 or φ_2 is a formula in* $\mathrm{Bool}[\Sigma]$*,*
- *$\wedge$-restricted, if for every subformula $\varphi_1 \wedge \varphi_2$ of ψ:*
  - *the sub-formula φ_1 is a formula in* $\mathrm{aBool}[\Sigma, \mathbb{M}]$*, or*
  - *φ_2 is a formula in* $\mathrm{Bool}[\Sigma]$*,*
- *commutatively $\wedge$-restricted, if for every subformula $\varphi_1 \wedge \varphi_2$ of ψ:*
  - *the sets* $\mathrm{Const}(\varphi_1)$ *and* $\mathrm{Const}(\varphi_2)$ *commute, or*
  - *φ_1 is a formula in* $\mathrm{aBool}[\Sigma, \mathbb{M}]$*.*

Note that if ψ is strongly $\wedge$-restricted, then it is $\wedge$-restricted. If ψ is $\wedge$-restricted, it is commutatively $\wedge$-restricted.

We introduce the following notations for these fragments of $\mathrm{wEMSO}[\Sigma, \mathbb{M}]$:

- Let $\mathrm{wEMSO}^{\forall,\wedge_s,d}[\Sigma, \mathbb{M}] \subseteq \mathrm{wEMSO}[\Sigma, \mathbb{M}]$ denote the set of all $\forall$-restricted, strongly $\wedge$-restricted and M-restricted formulas.
- Let $\mathrm{wEMSO}^{\forall,\wedge_s}[\Sigma, \mathbb{M}] \subseteq \mathrm{wEMSO}[\Sigma, \mathbb{M}]$ denote the set of all $\forall$-restricted and strongly $\wedge$-restricted formulas.
- Let $\mathrm{wEMSO}^{\forall,\wedge}[\Sigma, \mathbb{M}] \subseteq \mathrm{wEMSO}[\Sigma, \mathbb{M}]$ denote the set of all $\forall$-restricted and $\wedge$-restricted formulas.
- Let $\mathrm{wEMSO}^{\forall,\wedge_c}[\Sigma, \mathbb{M}] \subseteq \mathrm{wEMSO}[\Sigma, \mathbb{M}]$ denote the set of all $\forall$-restricted and commutatively $\wedge$-restricted formulas.

Our main result of this section will be the following:

Theorem 1.37. *Let $\mathbb{M}$ be a ppv-monoid and let $\mathbb{L} : \Sigma^{++} \to M$ be a picture series.*

1. *$\mathbb{L}$ is* W2OTA*-recognizable if and only if $\mathbb{L} = [\![\psi]\!]$ for a formula $\psi \in \mathrm{wEMSO}^{\forall,\wedge_s,d}[\Sigma, \mathbb{M}]$.*
2. *Let $\mathbb{M}$ be regular. Then $\mathbb{L}$ is* W2OTA*-recognizable if and only if $\mathbb{L} = [\![\psi]\!]$ for a formula $\psi \in \mathrm{wEMSO}^{\forall,\wedge_s}[\Sigma, \mathbb{M}]$.*
3. *Let $\mathbb{M}$ be left-distributive. Then $\mathbb{L}$ is* W2OTA*-recognizable if and only if $\mathbb{L} = [\![\psi]\!]$ for a formula $\psi \in \mathrm{wEMSO}^{\forall,\wedge}[\Sigma, \mathbb{M}]$.*

4. *Let $\mathbb{M}$ be cc-picture valuation semiring. Then $\mathbb{L}$ is* W2OTA*-recognizable if and only if $\mathbb{L} = [\![\psi]\!]$ for a formula $\psi \in \mathrm{wEMSO}^{\forall,\wedge_c}[\Sigma, \mathbb{M}]$.*

To prove this result, first by induction over the structure of ψ we will build a W2OTA $\mathcal{A}_\psi$ recognizing $[\![\psi]\!]$. We let M_ψ be the smallest subset of M containing $\mathrm{Const}(\psi) \cup \{\mathbf{0}, \mathbf{1}\}$ which is closed under the operations $+$ and $\diamond$. The weights of the transitions of $\mathcal{A}_\psi$ are taken from M_ψ. For the converse we use a different method from the one applied in [52]. In [52], first a weighted 2-dimensional on-line tessellation automaton was converted into another device called weighted picture automaton, and then it was shown that picture series recognized by this weighted device are definable. Here, we only consider W2OTA and we show that picture series recognized by this weighted automaton device are definable in terms of formulas in our logic.

Proposition 1.38. *Let $\mathbb{M}$ be regular and $\varphi \in \mathrm{Bool}[\Sigma]$ or $\varphi = d$ for some $d \in M$. Then $[\![\varphi]\!]$ is* W2OTA*-recognizable over $\mathrm{Const}(\varphi) \cup \{\mathbf{1}\}$.*

Proof. Let $\varphi \in \mathrm{Bool}[\Sigma]$. Then φ can be regarded as a classical boolean first-order formula defining the picture language $L(\varphi)$, which is unambiguous, due to Proposition 1.29. We know that $[\![\varphi]\!] = \mathbf{1}_{L(\varphi)}$. Let $\mathcal{A} = (Q, T, I, F)$ be an unambiguous 2OTA recognizing $L(\varphi)$. Then, by Lemma 1.25, $\mathcal{A}^1 = (Q, T, I, F, \mathrm{wt})$ is the W2OTA which recognizes $[\![\varphi]\!]$ and for which $\mathrm{wt}(T) \subseteq \{\mathbf{1}\}$. Next, let $\varphi = d$ for some $d \in M$. Then $[\![d]\!] = d\mathbf{1}_{\Sigma^{++}}$ is W2OTA-recognizable over $\{d, \mathbf{1}\}$, since we assumed that $\mathbb{M}$ to be regular. □

Proposition 1.39. *Let $\varphi \in \mathrm{aBool}[\Sigma, \mathbb{M}]$. Then $[\![\varphi]\!](p) \in M_\varphi$ for every $p \in \Sigma^{++}$.*

Proof. For $\varphi \in \mathrm{Bool}[\Sigma]$ or $\varphi = d$ the claim is clear by Proposition 1.38. Now let $\varphi, \varphi' \in \mathrm{aBool}[\Sigma, \mathbb{M}]$ and $\mathcal{V} = \mathit{free}(\varphi) \cup \mathit{free}(\varphi')$. Due to Lemma 1.33 and Proposition 1.35 we have $[\![\varphi]\!]_\mathcal{V} = \bigoplus_{i=1}^m d_i \mathbf{1}_{L_i}$ and $[\![\varphi']\!]_\mathcal{V} = \bigoplus_{j=1}^n d'_j \mathbf{1}_{L'_j}$ are FO step functions. Now by induction hypothesis, $d_i \in M_\varphi$ and $d'_j \in M_{\varphi'}$, for all $i \in \{1, 2, ..., m\}$ and $j \in \{1, 2, ..., n\}$. Now we have:

$$[\![\varphi \vee \varphi']\!] = [\![\varphi]\!] + [\![\varphi']\!] = \bigoplus_{i=1}^{m} \bigoplus_{j=1}^{n} (d_i + d'_j)\mathbf{1}_{L_i \cap L'_j},$$

$$[\![\varphi \wedge \varphi']\!] = [\![\varphi]\!] \diamond [\![\varphi']\!] = \bigoplus_{i=1}^{m} \bigoplus_{j=1}^{n} (d_i \diamond d'_j)\mathbf{1}_{L_i \cap L'_j}.$$

Hence, for every $p \in \Sigma^{++}$ we obtain $[\![\varphi \vee \varphi']\!](p)$, $[\![\varphi \wedge \varphi']\!](p) \in M_{\varphi \vee \varphi'} = M_{\varphi \wedge \varphi'}$, which completes the proof. □

Proposition 1.40. *Let $\mathbb{M}$ be a ppv-monoid, $E, E' \subseteq M$ and $\varphi, \varphi' \in \mathrm{wFO}[\Sigma, \mathbb{M}]$ such that $[\![\varphi]\!]$ is* W2OTA*-recognizable over E and $[\![\varphi']\!]$ is* W2OTA*-recognizable over E'. Then $[\![\varphi \vee \varphi']\!]$ is* W2OTA*-recognizable over $E \cup E'$.*

Proof. Let $\mathcal{V} = \mathit{free}(\varphi) \cup \mathit{free}(\varphi')$. By Proposition 1.27, $[\![\varphi]\!]_{\mathcal{V}}$ is W2OTA-recognizable over E and $[\![\varphi']\!]_{\mathcal{V}}$ is W2OTA-recognizable over E'. Now by Proposition 1.14, we have $[\![\varphi \vee \varphi']\!] = [\![\varphi]\!]_{\mathcal{V}} + [\![\varphi']\!]_{\mathcal{V}}$ is W2OTA-recognizable over $E \cup E'$. □

Proposition 1.41. *Let $\mathbb{M}$ be a ppv-monoid, $E \subseteq M$, and $\varphi \in \mathrm{wFO}[\Sigma, \mathbb{M}]$ such that $[\![\varphi]\!]$ is* W2OTA*-recognizable over E. Then $[\![\exists x\ \varphi]\!]$ and $[\![\exists X\ \varphi]\!]$ are* W2OTA*-recognizable over E, as well.*

Proof. We give the proof for the existential first-order quantification case. Let $\mathcal{V} = \mathit{free}(\exists x\ \varphi)$. Note that $x \notin \mathcal{V}$. Consider the projection π from $\Sigma^{++}_{\mathcal{V} \cup \{x\}}$ to $\Sigma^{++}_{\mathcal{V}}$ which erases the x-level. Then we have:

$$\begin{aligned}
[\![\exists x\ \varphi]\!](p, \sigma) &= \sum([\![\varphi]\!](p, \sigma[x/(i,j)]) \mid (i,j) \in \mathrm{dom}(p)) \\
&= \sum([\![\varphi]\!](p, \sigma') \mid \pi(p, \sigma') = (p, \sigma)) \\
&= \pi([\![\varphi]\!])(p, \sigma).
\end{aligned}$$

Due to Proposition 1.11(1), $[\![\exists x\ \varphi]\!]$ is W2OTA-recognizable over E. The proof for the second-order variable case is similar. □

Proposition 1.42. *Let $\mathbb{M}$ be a ppv-monoid, $E, E' \subseteq M$, and $\varphi, \varphi' \in \mathrm{wFO}[\Sigma, \mathbb{M}]$.*

1. *If $[\![\varphi]\!]$ is* W2OTA*-recognizable over E and $\varphi' \in \mathrm{Bool}[\Sigma]$, then $[\![\varphi \wedge \varphi']\!]$ and $[\![\varphi' \wedge \varphi]\!]$ are* W2OTA*-recognizable over E.*
2. *Let $\mathbb{M}$ be left-distributive. If $\varphi \in \mathrm{aBool}[\Sigma, \mathbb{M}]$ and $[\![\varphi']\!]$ is* W2OTA*-recognizable over E', then $[\![\varphi \wedge \varphi']\!]$ is* W2OTA*-recognizable over $\mathrm{im}[\![\varphi]\!] \diamond E'$.*
3. *Let $\mathbb{M}$ be a cc-picture valuation semiring, $[\![\varphi]\!]$ be* W2OTA*-recognizable over E, $[\![\varphi']\!]$ be* W2OTA*-recognizable over E'. If E and E' commute, then $[\![\varphi \wedge \varphi']\!]$ is* W2OTA*-recognizable over $E \diamond E'$.*

Proof. Let $\mathcal{V} = \mathit{free}(\varphi) \cup \mathit{free}(\varphi')$.
1. Since $\varphi' \in \mathrm{Bool}[\Sigma]$, $[\![\varphi']\!]_{\mathcal{V}} = \mathbf{1}_{L_{\mathcal{V}}(\varphi')}$ where $L_{\mathcal{V}}(\varphi') \subseteq \Sigma^{++}_{\mathcal{V}}$. Due to Proposition 1.29, $L_{\mathcal{V}}(\varphi')$ is unambiguously 2OTA-recognizable. Let

$\mathbb{L} = [\![\varphi]\!]_{\mathcal{V}}$. By Proposition 1.27, $\mathbb{L}$ is W2OTA-recognizable over E. Now by Lemma 1.15(1), $[\![\varphi' \wedge \varphi]\!] = [\![\varphi \wedge \varphi']\!] = [\![\varphi]\!] \diamond [\![\varphi']\!] = \mathbb{L} \diamond \mathbf{1}_{L_{\mathcal{V}}(\varphi')}$ is W2OTA-recognizable over E.

2. Since $\varphi \in \mathrm{aBool}[\Sigma, \mathbb{M}]$, then due to Proposition 1.35 and Lemma 1.33, $\mathbb{L} = [\![\varphi]\!]_{\mathcal{V}} = \bigoplus_{i=1}^{n} d_i \mathbf{1}_{L_i}$ is an FO step function where $(L_i)_{i=1,...,n}$ is a partition of FO-definable picture languages over $\Sigma_{\mathcal{V}}^{++}$. By Proposition 1.27, $\mathbb{L}' = [\![\varphi']\!]_{\mathcal{V}}$ is W2OTA-recognizable over E'. Now, $[\![\varphi \wedge \varphi']\!] = \mathbb{L} \diamond \mathbb{L}'$ is W2OTA-recognizable over $\mathrm{im}([\![\varphi]\!]) \diamond E'$, by Lemma 1.31.

3. By Proposition 1.27, $\mathbb{L} = [\![\varphi]\!]_{\mathcal{V}}$ and $\mathbb{L}' = [\![\varphi']\!]_{\mathcal{V}}$ are W2OTA-recognizable over E and E', respectively. Now due to Lemma 1.32, $[\![\varphi \wedge \varphi']\!] = \mathbb{L} \diamond \mathbb{L}'$ is W2OTA-recognizable over $E \diamond E'$.

□

Proposition 1.43. *Let $\mathbb{M}$ be a ppv-monoid and let $\varphi \in \mathrm{aBool}[\Sigma, \mathbb{M}]$. Then $[\![\forall x\, \varphi]\!]$ is* W2OTA*-recognizable over* $\mathrm{im}([\![\varphi]\!])$.

Proof. Due to Proposition 1.35, $[\![\varphi]\!] = \bigoplus_{k=1}^{n} d_k 1_{L_k}$ is an FO step function where $(L_k)_{k=1,...,n}$ is a partition of FO-definable picture languages over Σ_{φ}^{++}. Let $\mathcal{V} = \mathit{free}(\varphi) \backslash \{x\}$. We proceed as in [41, 52]. Let $\widetilde{\Sigma} = \Sigma \times \{1, 2, ..., n\}$. A picture in $(\widetilde{\Sigma}_{\mathcal{V}})^{++}$ is written as a tuple (p, ν, σ) where $(p, \sigma) \in \Sigma_{\mathcal{V}}^{++}$ and $\nu \in \{1, 2, ..., n\}^{\mathrm{dom}(p)}$ (ν is interpreted as a mapping from $\mathrm{dom}(p)$ to $\{1, 2, ..., n\}$). Let $\widetilde{L}$ be the picture language of all $(p, \nu, \sigma) \in (\widetilde{\Sigma}_{\mathcal{V}})^{++}$ such that (p, σ) is valid and for all $(i, j) \in \mathrm{dom}(p)$ and $k \in \{1, 2, ..., n\}$ we have :

$$\nu(i, j) = k \;\Leftrightarrow\; (p, \sigma[x/(i, j)]) \in L_k.$$

Observe that for each $(p, \sigma) \in \Sigma_{\mathcal{V}}^{++}$ there is a unique $\nu \in \{1, 2, ..., n\}^{\mathrm{dom}(p)}$ such that $(p, \nu, \sigma) \in \widetilde{L}$, because $(L_k)_{k=1,...,n}$ form a partition of Σ_{φ}^{++}. In [52] it was proved that $\widetilde{L}$ is FO-definable and therefore there exists an unambiguous 2OTA $\widetilde{\mathcal{A}} = (Q, T, I, F)$ over $\widetilde{\Sigma}_{\mathcal{V}}$ computing $\widetilde{L}$. Now from $\widetilde{\mathcal{A}}$ we obtain an unambiguous W2OTA $\mathcal{A} = (Q, T, I, F, \mathrm{wt})$ by putting $\mathrm{wt}(q_h, q_v, (a, k, s), q) = d_k$ for $(q_h, q_v, (a, k, s), q) \in T$ where $q_h, q_v, q \in Q$ and $(a, k, s) \in \widetilde{\Sigma}_{\mathcal{V}}$. We recall that $[\![\varphi]\!] = \bigoplus_{k=1}^{n} d_k 1_{L_k}$ and so $\mathrm{wt}(T) \subseteq \mathrm{im}([\![\varphi]\!])$. Since $\widetilde{\mathcal{A}}$ is unambiguous, over a picture (p, ν, σ) there is at most one unique successful run $\rho = (c_{i,j})_{(i,j) \in \mathrm{dom}(p)}$ in $\widetilde{\mathcal{A}}$ which is evaluated in $\mathcal{A}$ as follows:

$$[\![\mathcal{A}]\!](p, \nu, \sigma) = \mathrm{wt}(\rho) = \mathrm{val}(\mathrm{wt}(c_{i,j})_{(i,j) \in \mathrm{dom}(p)})$$

and if $(p, \nu, \sigma) \notin \widetilde{L}$ then $[\![\mathcal{A}]\!](p, \nu, \sigma) = \mathbf{0}$. If $(p, \nu, \sigma) \in \widetilde{L}$, then for every $(i, j) \in \mathrm{dom}(p)$: $\nu(i, j) = k$ implies that $\mathrm{wt}(c_{i,j}) = d_k$. In

addition $(p, \sigma[x/(i,j)]) \in L_k$ and $[\![\varphi]\!](p, \sigma[x/(i,j)]) = d_k$. Now we apply the projection $\pi : \widetilde{\Sigma}_{\mathcal{V}} \to \Sigma_{\mathcal{V}}$, erasing the $\{1, ..., n\}$-level, which is defined by $\pi(a, k, s) = (a, s)$. Then for every valid $(p, \sigma) \in \Sigma_{\mathcal{V}}^{++}$ and the unique ν such that $(p, \nu, \sigma) \in \widetilde{L}$, we get

$$\begin{aligned}\pi([\![\mathcal{A}]\!])(p, \sigma) &= [\![\mathcal{A}]\!](p, \nu, \sigma) = \mathrm{val}(\mathrm{wt}(c_{i,j})_{(i,j)\in \mathrm{dom}(p)}) \\ &= \mathrm{val}([\![\varphi]\!](p, \sigma[x/(i,j)])_{(i,j)\in \mathrm{dom}(p)}) \\ &= [\![\forall x\ \varphi]\!](p, \sigma).\end{aligned}$$

Therefore, $[\![\forall x\ \varphi]\!] = \pi([\![\mathcal{A}]\!])$ is W2OTA-recognizable over $\mathrm{im}([\![\varphi]\!])$, due to Proposition 1.11(1). □

Propositions 1.38 - 1.43 will give the following result.

Theorem 1.44. *Let $\mathbb{M}$ be a ppv-monoid and $\psi \in \mathrm{wEMSO}[\Sigma, \mathbb{M}]$. Assume that one of the following conditions holds:*

1. *ψ is a formula in $\mathrm{wEMSO}^{\forall, \wedge_s, d}[\Sigma, \mathbb{M}]$,*
2. *$\mathbb{M}$ is regular and ψ is a formula in $\mathrm{wEMSO}^{\forall, \wedge_s}[\Sigma, \mathbb{M}]$,*
3. *$\mathbb{M}$ is left-distributive and ψ is a formula in $\mathrm{wEMSO}^{\forall, \wedge}[\Sigma, \mathbb{M}]$,*
4. *$\mathbb{M}$ is cc-picture valuation semiring and ψ is a formula in $\mathrm{wEMSO}^{\forall, \wedge_c}[\Sigma, \mathbb{M}]$.*

Then $[\![\psi]\!]$ is W2OTA*-recognizable over M_ψ.*

Proof. We proceed by induction over the structure of ψ. If $\psi \in \mathrm{Bool}[\Sigma]$ or $\psi = d$ for some $d \in M$, the result follows from regularity of $\mathbb{M}$ and Proposition 1.38. Note that in case (1), the formula ψ is M-restricted and therefore it does not include $\psi = d$ for some $d \in M$. If $\psi = \varphi_1 \vee \varphi_2$, $\psi = \exists x\ \varphi$, and $\psi = \exists X\ \varphi$ we apply Proposition 1.40 and 1.41. For $\psi = \varphi_1 \wedge \varphi_2$ we use the respective assumptions on $\mathbb{M}$ and ψ. In the first two cases, the result follows by Proposition 1.42, parts (1) and (2). Now assume that $\mathbb{M}$ is a cc-picture valuation semiring and $\psi = \varphi_1 \wedge \varphi_2$ is commutatively $\wedge$-restricted. First, if $\varphi_1 \in \mathrm{aBool}[\Sigma, \mathbb{M}]$, then we apply again Proposition 1.42, part (2). Otherwise, $\mathrm{Const}(\varphi_1)$ and $\mathrm{Const}(\varphi_2)$ commute. By induction hypothesis, $[\![\varphi_1]\!]$ and $[\![\varphi_2]\!]$ are W2OTA-recognizable over M_{φ_1} and M_{φ_2}, respectively. Since $\mathrm{Const}(\varphi_1)$ and $\mathrm{Const}(\varphi_2)$ commute and $(M, +, \diamond, \mathbf{0}, \mathbf{1})$ is a semiring, M_{φ_1} and M_{φ_2} also commute. Now if we apply Proposition 1.42, part (3), we have $[\![\varphi_1 \wedge \varphi_2]\!]$ is W2OTA-recognizable over $M_{\varphi_1} \diamond M_{\varphi_2} \subseteq M_{\varphi_1 \wedge \varphi_2}$. Finally, if $\psi = \forall x\ \varphi$, then the formula $\varphi \in \mathrm{aBool}[\Sigma, \mathbb{M}]$, and the result follows by Propositions 1.39 and 1.43. □

Now we give the proof of Theorem 1.37.

Proof of Theorem 1.37. The "if"-statements are immediate by Theorem 1.44. For the converse, let $\mathcal{A} = (Q, T, I, F, \mathrm{wt})$ be a W2OTA. For each transition $(q_h, q_v, a, q) \in T$ we choose a second-order variable $X_{(q_h,q_v,a,q)}$ and put $\mathcal{V} = \{X_{(q_h,q_v,a,q)} \mid (q_h, q_v, a, q) \in T\}$. In addition, we let $\overline{X} = (X_1, ..., X_n)$ be an enumeration of $\mathcal{V}$. We wish to construct a $\forall$-restricted, M-restricted and strongly $\wedge$-restricted sentence α such that $[\![\alpha]\!] = [\![\mathcal{A}]\!]$. For every $d \in M$, we define the almost boolean first-order formula

$$((x \in X) \to d) := ((x \in X) \wedge d) \vee (x \notin X)$$

which has the semantics

$$[\![(x \in X) \to d]\!](p, \sigma) = \begin{cases} d & \text{if } \sigma(x) \in \sigma(X), \\ \mathbf{1} & \text{otherwise.} \end{cases}$$

In fact, we have $[\![(x \in X) \to d]\!]_{\{x,X\}} = d.1_{\mathcal{L}((x \in X))} + \mathbf{1}.\mathbf{1}_{\mathcal{L}((x \notin X))}$ which is an FO step function. We also use the following formulas:

$$\begin{aligned} part(X_1, ..., X_n) &:= \forall x \, (\bigvee_{i=1,...,n} [x \in X_i \wedge \bigwedge_{j \neq i} \neg(x \in X_j)]), \\ \varphi_t(x) &:= \forall y \, \neg(y S_h x), \\ \varphi_l(x) &:= \forall y \, \neg(y S_v x), \\ \varphi_{br}(x) &:= \forall y \, [\neg(x S_h y) \wedge \neg(x S_v y)], \end{aligned}$$

$$\chi_t := \forall x \, ([\varphi_t(x) \wedge (\bigvee_{q_h^x, q^x \in Q, q_v^x \in I, a \in \Sigma} x \in X_{(q_h^x, q_v^x, a, q^x)})] \vee \exists s \, (s S_h x)),$$

$$\chi_l := \forall x \, ([\varphi_l(x) \wedge (\bigvee_{q_v^x, q^x \in Q, q_h^x \in I, a \in \Sigma} x \in X_{(q_h^x, q_v^x, a, q^x)})] \vee \exists s \, (s S_v x)),$$

$$\chi_{br} := \forall x \, ([\varphi_{br}(x) \wedge (\bigvee_{q_h^x, q_v^x \in Q, q^x \in F, a \in \Sigma} x \in X_{(q_h^x, q_v^x, a, q^x)})] \vee [\exists s \, ((x S_h s) \vee (x S_v s))].$$

The formulas φ_t, φ_l and φ_{br} are used to define the top and left borders and the position in the bottom-right corner, respectively. Then the formulas χ_t, χ_l and χ_{br} simulate accepting conditions of the automaton $\mathcal{A}$ at the top and left borders and bottom-right corner.

Now we define

$$\begin{aligned}
\psi(\overline{X}) := & \; part(\overline{X}) \\
& \wedge \bigwedge_{(q_h,q_v,a,q)\in T} \forall x \; ((x \in X_{(q_h,q_v,a,q)}) \rightarrow P_a(x)) \\
& \wedge \; \forall x \; \forall z \; ((xS_vz) \rightarrow \\
& \qquad \bigvee_{q_h^x,q_v^x,q_v^z,q^x,q^z\in Q;a,b\in\Sigma} (x \in X_{(q_h^x,q_v^x,a,q^x)}) \wedge (z \in X_{(q^x,q_v^z,b,q^z)})) \\
& \wedge \; \forall y \; \forall z \; ((yS_hz) \rightarrow \\
& \qquad \bigvee_{q_h^y,q_v^y,q_h^z,q^y,q^z\in Q;c,b\in\Sigma} (y \in X_{(q_h^y,q_v^y,c,q^y)}) \wedge (z \in X_{(q_h^z,q^y,b,q^z)})) \\
& \wedge \; \chi_t \; \wedge \; \chi_l \; \wedge \; \chi_{br}.
\end{aligned}$$

The formula ψ simulates unweighted runs in the automaton $\mathcal{A}$ as pictures of transitions in T. The first line states that for every pixel the automaton applies exactly one transition reading that pixel. The next two lines describe the conditions concerning the inner rules in which a run of a 2OTA satisfies. The last line simulates that successful runs have to fulfil the accepting conditions. Next we can construct a boolean first-order formula $\psi^+(\overline{X})$ with $[\![\psi^+(\overline{X})]\!] = 1_{L(\psi(\overline{X}))}$. The boolean formula $\psi^+(\overline{X})$ has for a picture p and an assignment σ value $\mathbf{1}$ if $\psi[\sigma(\overline{X})]$ describes a successful run of $\mathcal{A}$ on p. Otherwise, it takes the value $\mathbf{0}$. Now we put

$$\varphi(\overline{X}) \; := \psi^+(\overline{X}) \; \wedge \; \forall x \bigwedge_{(q_h,q_v,a,q)\in T} (x \in X_{(q_h,q_v,a,q)}) \rightarrow \mathrm{wt}(q_h, q_v, a, q).$$

Whenever $\psi[\sigma(\overline{X})]$ defines a successful run ρ of $\mathcal{A}$ on p, then

$$[\![\varphi(\overline{X})]\!](p, \sigma) = \mathrm{wt}(\rho),$$

otherwise the formula evaluates to $\mathbf{0}$. Note that $\varphi(\overline{X})$ is $\forall$-restricted, M-restricted and strongly $\wedge$-restricted.

Finally, we define the $\forall$- , M- and strongly $\wedge$-restricted sentence α

$$\alpha \; := \exists X_1 ... \exists X_n \varphi(X_1, ..., X_n)$$

for which we have $[\![\alpha]\!] = [\![\mathcal{A}]\!]$.

□

Chapter 2

$+\omega$-Picture Languages Recognizable by Büchi-Tiling Systems

In this chapter, we generalize the notion of finite pictures to $+\omega$-*pictures*, i.e., pictures which have finite number of rows and infinite number of columns. We extend conventional tiling systems with a Büchi acceptance condition in order to define the class of *Büchi-tiling recognizable* $+\omega$-*picture languages*. The class of recognizable $+\omega$-picture languages is indeed, a natural generalization of ω-regular languages. We show that this new class of recognizable picture languages has the same closure properties as the class of tiling recognizable languages of finite pictures [64]. We characterize the class of Büchi-tiling recognizable $+\omega$-picture languages by generalized Büchi-tiling systems and by EMSO$^{\infty}$ logic, i.e., an extension of existential monadic second-order logic with quantification of infinite sets. Additioanlly, we extend many results and techniques, which have been investigated for tiling recognizable languages of finite pictures or for ω-regular languages recognized by Büchi-automata, to the class of Büchi-tiling recognizable $+\omega$-picture languages. However, we show that the Büchi characterization theorem, which states that the ω-regular languages are finite unions of languages of the form $L_1 \cdot L_2^{\omega}$, for regular languages L_1 and L_2, cannot be extented from regular ω-languages to Büchi-tiling recognizable languages of $+\omega$-pictures.

2.1 $+\omega$-Pictures

Recall that $\mathbb{N}_{\geqslant 1} = \mathbb{N}\backslash\{0\}$ where $\mathbb{N} = \{0, 1, 2, \cdots\}$, and $[m]$ denotes the set $\{1, 2, \cdots, m\}$, for any $m \in \mathbb{N}_{\geqslant 1}$. Let Σ be an alphabet. For $m \in \mathbb{N}_{\geqslant 1}$, an *$m\omega$-picture* over Σ is a mapping $p : [m] \times \mathbb{N}_{\geqslant 1} \to \Sigma$, i.e., p has exactly m rows and an infinite number of columns. We write $\Sigma^{m\omega}$ to denote the set of all $m\omega$-pictures over Σ. The set of all *$+\omega$-pictures* over Σ is the set $\Sigma^{+\omega} := \bigcup_{m\in\mathbb{N}_{\geqslant 1}} \Sigma^{m\omega}$. A *$+\omega$-picture language* over Σ is a subset of $\Sigma^{+\omega}$. Recall that, the number $\ell_v(p) := m$ of rows is called the *height* of p. We write $\ell_h(p) = \omega$ to indicate that there is an infinite number of columns. The *size* of p is (m, ω).

We identify finite pictures of height 1 over Σ with finite non-empty words over Σ (i.e., with elements in Σ^+), and we identify $+\omega$-pictures of height 1 over Σ with ω-words over Σ (i.e., with elements in Σ^ω). When given a (finite or $+\omega$-)picture p and numbers j_1, j_2 with $1 \leqslant j_1 \leqslant j_2 \leqslant \ell_h(p)$ we write $p[j_1, j_2]$ for the picture obtained from p by deleting all columns j with $j < j_1$ or $j > j_2$.

The *column concatenation* of a finite picture p and a (finite or $+\omega$-)picture q of the same height $m = \ell_v(p) = \ell_v(q)$ yields the picture $p \circledcirc q$ of height m whose first $\ell_h(p)$ columns are identical with p and whose remaining columns $\ell_h(p)+1$, $\ell_h(p)+2$, ... are identical with the columns 1, 2, ... of q, i.e., $(p \circledcirc q)[1, \ell_v(p)] = p$ and $(p \circledcirc q)[\ell_v(p)+1, \ell_v(p)+\ell_h(p)] = q$ (using the convention "$n+\omega = \omega$"). The column concatenation can be illustrated as follows; if

$$p = \begin{matrix} p_{1,1} & \cdots & p_{1,n} \\ \vdots & \ddots & \vdots \\ p_{m,1} & \cdots & p_{m,n} \end{matrix}, \quad q = \begin{matrix} q_{1,1} & q_{1,2} & \cdots \\ \vdots & \vdots & \ddots \\ q_{m,1} & q_{m,2} & \cdots \end{matrix},$$

then

$$p \circledcirc q = \begin{matrix} p_{1,1} & \cdots & p_{1,n} & q_{1,1} & q_{1,2} & \cdots \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots \\ p_{m,1} & \cdots & p_{m,n} & q_{m,1} & q_{m,2} & \cdots \end{matrix}.$$

The concatenation of a language $L \subseteq \Sigma^{++}$ of finite pictures and a language $L' \subseteq \Sigma^{+\omega}$ of $+\omega$-pictures is defined as

$$L \circledcirc L' = \{\, p \circledcirc p' \,:\, p \in L,\ p' \in L',\ \ell_v(p) = \ell_v(p') \,\},$$

which means a picture q belongs to $L \circledcirc L'$ if, and only if, some initial segment of q belongs to L and the rest of q belongs to L'. In case $L, L' \subseteq \Sigma^{++}$ are

two languages of finite pictures, the concatenation $L \oplus L'$ is the picture language

$$L \oplus L' = \{p \oplus p' : p \in L,\ p' \in L',\ \ell_v(p) = \ell_v(p')\}.$$

We define the iterated concatenation $L^{\oplus n}$ via $L^{\oplus 1} = L$ and $L^{\oplus n} = L \oplus L^{\oplus(n-1)}$ for every $n \in \mathbb{N}$ with $n \geqslant 2$. Clearly, for every $n \in \mathbb{N}_{\geqslant 1}$, $L^{\oplus n}$ is a language of finite pictures. We let $L^{\oplus\omega}$ be the $+\omega$-picture language that consists of all $+\omega$-pictures p for which there is an infinite sequence $1 = j_1 < j_2 < \cdots$ of integers such that for every $i \in \mathbb{N}_{\geqslant 1}$, the picture $p[j_i, j_{i+1}-1]$ belongs to L. That means, $L^{\oplus\omega}$ consists of all $+\omega$ pictures of the form $p_1 \oplus p_2 \oplus p_3 \oplus \cdots$ where $p_i \in L$ for every $i \in \mathbb{N}_{\geqslant 1}$, and all p_i have the same height. For a finite picture $p \in \Sigma^{++}$ we write $p^{\oplus\omega}$ for the unique $+\omega$-picture in $\{p\}^{\oplus\omega}$. Accordingly, for $n \in \mathbb{N}_{\geqslant 1}$ we write $p^{\oplus n}$ for the unique picture in the set $\{p\}^{\oplus n}$.

2.2 Büchi-Tiling Recognizable $+\omega$-Picture Languages

Local sets of words play an important role in the theory of regular string languages. This notion has been generalized to languages of finite pictures [61] and to $\omega\omega$-picture languages [35]. In this section, we extend this notion to $+\omega$-picture languages and we introduce tiling systems with a Büchi acceptance condition. We show that *Büchi-tiling systems* are strictly stronger than tiling systems. In addition, we introduce *generalized Büchi-tiling system* and we show that Büchi-tiling systems are equivalent to generalized Büchi tiling systems.

For the convenience of the presentation, we introduce the following notions. For a (finite or $+\omega$-)picture p over Σ and for numbers i_1, i_2, j_1, j_2 with $1 \leqslant i_1 \leqslant i_2 \leqslant \ell_v(p)$ and $1 \leqslant j_1 \leqslant j_2 \leqslant \ell_h(p)$ we write $p_{i_2}^{i_1}[j_1, j_2]$ for the subpicture of p at rows $i_1, \ldots, i_2$ and columns $j_1, \ldots, j_2$, i.e., $p_{i_2}^{i_1}[j_1, j_2]$ is the picture obtained from p by deleting all rows i with $i < i_1$ or $i > i_2$ and deleting all columns j with $j < j_1$ or $j > j_2$, which is illustrated as follows:

$$p_{i_2}^{i_1}[j_1, j_2] \quad = \quad \begin{array}{ccc} p_{i_1,j_1} & \cdots & p_{i_1,j_2} \\ \vdots & \ddots & \vdots \\ p_{i_2,j_1} & \cdots & p_{i_2,j_2} \end{array} .$$

For numbers $m, n \in \mathbb{N}_{\geqslant 1}$ we write $T_{m,n}(p)$ for the set of all subpictures of

p of size (m,n), i.e.,

$$T_{m,n}(p) \;=\; \Big\{ p^i_{i+(m-1)}[j, j+(n-1)] \;:\; 1 \leqslant i \leqslant \ell_v(p)-m,\; 1 \leqslant j \leqslant \ell_h(p)-n \Big\}.$$

(with the convention "$\omega - n = \omega$"). For an alphabet Γ, we write $\hat{\Gamma}$ for the alphabet $\Gamma \cup \{\#\}$, where $\#$ is a special boundary symbol that does not belong to Γ. For a finite picture q of size (m,n) over Γ, we write $\hat{q}$ for the picture of size $(m{+}2, n{+}2)$ over $\hat{\Gamma}$, obtained by surrounding q with the boundary symbol $\#$. Accordingly, for a $+\omega$-picture q over Γ we write $\hat{q}$ for the $(m{+}2)\omega$-picture over $\hat{\Gamma}$, obtained by surrounding q with the boundary symbol $\#$ from the left, top and bottom. A *tile* is a picture of size $(2,2)$ over the alphabet $\hat{\Gamma}$.

Definition 2.1. *Let Γ be an alphabet and let $\bullet$ be one of the symbols $+$ or ω. The $+\bullet$-picture language* recognized *by a set $\Theta \subseteq \hat{\Gamma}^{2,2}$ of tiles is*

$$L^{+\bullet}(\Theta) := \{\; q \in \Gamma^{+\bullet} \;:\; T_{2,2}(\hat{q}) \subseteq \Theta \;\}.$$

A $+\bullet$-picture language L over Γ is called local if there exists a set $\Theta \subseteq \hat{\Gamma}^{2,2}$ of tiles such that $L = L^{+\bullet}(\Theta)$.

Note that $L^{++}(\Theta)$ is the set of all finite pictures q over Γ, such that every subpicture of $\hat{q}$ of size $(2,2)$ belongs to Θ. Similarly, $L^{+\omega}(\Theta)$ is the set of all $+\omega$-pictures q over Γ, such that every subpicture of $\hat{q}$ of size $(2,2)$ belongs to Θ.

Recall that a *projection* is the mapping $\pi : \Gamma \to \Sigma$, for alphabets Γ and Σ which can be lifted to pictures and picture languages in the canonical way: For a picture q over Γ, $\pi(q)$ is the picture p over Σ of the same height and width as q, where for each row i and each column j, the letter p_{ij} in row i and column j is $\pi(q_{ij})$. For a picture language L over Γ, we let $\pi(L) = \{\pi(q) \;:\; q \in L\}$.

Definition 2.2. *A language $L \subseteq \Sigma^{++}$ of finite pictures is tiling recognizable if there exists an alphabet Γ, a local picture language L' over Γ, and a projection $\pi : \Gamma \to \Sigma$, such that $L = \pi(L')$.*

When dealing with recognizability, it is often convenient to assume that the alphabet Γ has the special form $\Gamma = \Sigma \times Q$, and the projection $\pi : \Gamma \to \Sigma$ just cancels the Q-component. It is straighforward to see that this assumption can be made without loss of generality (cf., e.g., [64]). Thus, a language $L \subseteq \Sigma^{++}$ is tiling recognizable if, and only if, there exists a *tiling system* $\mathcal{T}$ with $L = L(\mathcal{T})$ in the following sense.

Definition 2.3. *A tiling system is a 3-tuple $\mathcal{T} = (\Sigma, Q, \Theta)$ where Σ and Q are sets of finite alphabets, and $\Theta \subseteq \hat{\Gamma}^{2,2}$ for $\Gamma = \Sigma \times Q$. The elements of Q are called* states *of $\mathcal{T}$.*

Let Σ and Q be alphabets. For a (finite or $+\omega$-)picture p over Σ and a picture r over Q of the same size as p, we write $(p \times r)$ for the picture q over $\Gamma = \Sigma \times Q$ that has the same size as p, and where for every row i and every column j, the entry q_{ij} in row i and column j is (p_{ij}, r_{ij}). Let $\mathcal{T} = (\Sigma, Q, \Theta)$ be a tiling system, let $\bullet$ be one of the symbols $+$ or ω, and let p be a $+\bullet$-picture over Σ. We describe the behaviour of $\mathcal{T}$ as follows: A *run* of $\mathcal{T}$ on p is a picture r over Q of the same size as p, such that the picture $q = (p \times r)$ has the following property: every subpicture of size $(2,2)$ of $\hat{q}$ belongs to Θ, i.e., $T_{2,2}(\hat{q}) \subseteq \Theta$. The picture p is *accepted* by $\mathcal{T}$ if there exists a run of $\mathcal{T}$ on p. The $+\bullet$-picture language *recognized* by $\mathcal{T}$ is the set $L^{+\bullet}(\mathcal{T})$ of all $+\bullet$-pictures p over Σ that are accepted by $\mathcal{T}$. A picture language $L \subseteq \Sigma^{+\bullet}$ is *tiling recognizable* if there is a tiling system $\mathcal{T}$ with $L = L^{+\bullet}(\mathcal{T})$.

We now extend tiling systems with a *Büchi acceptance condition*. This will lead to a notion of *Büchi-tiling recognizable* $+\omega$-picture languages that can be viewed as a 2-dimensional generalization of the ω-regular languages (i.e., the languages of ω-words recognized by Büchi-automata).

Definition 2.4. *A Büchi-tiling system is a 4-tuple $\mathcal{S} = (\Sigma, Q, \Theta, F)$, where (Σ, Q, Θ) is a tiling system and $F \subseteq Q$. The elements of F are called* accepting states*; the set F is called the* acceptance condition*.*

Let $\mathcal{S} = (\Sigma, Q, \Theta, F)$ be a Büchi-tiling system and let p be a $+\omega$-picture over Σ. We describe the behaviour of $\mathcal{S}$ as follows: A *run* of $\mathcal{S}$ on p is a run of the tiling system $\mathcal{T} = (\Sigma, Q, \Theta)$ on p. For a run r of $\mathcal{S}$ on p we write $\inf_1(r)$ for the set of states that occur infinitely often in the first row of r. A run r of $\mathcal{S}$ on p is *accepting* if $\inf_1(r) \cap F \neq \emptyset$, i.e., there is an accepting state that occurs infinitely often in the first row of the run. A $+\omega$-picture p over Σ is *accepted* by $\mathcal{S}$ if there exists an accepting run of $\mathcal{S}$ on p. The $+\omega$-picture language *recognized* by $\mathcal{S}$ is the set $L^{+\omega}(\mathcal{S})$ of all $+\omega$-pictures p over Σ that are accepted by $\mathcal{S}$. A $+\omega$-picture language L over Σ is *Büchi-tiling recognizable* if there is a Büchi-tiling system $\mathcal{S} = (\Sigma, Q, \Theta, F)$ with $L = L(\mathcal{S})$.

A simple counting argument shows that Büchi-tiling systems are strictly stronger than tiling systems:

Proposition 2.5. *Let $\Sigma = \{a, b\}$ consist of two distinct letters. The $+\omega$-picture language L over Σ which consists of all $p \in \Sigma^{1\omega}$ that contain an infinite number of a's, is Büchi-tiling recognizable, but not tiling recognizable.*

Proof. The Büchi-tiling recognizability of L is straightforward. Assume for contradiction that L is also tiling recognizable, and let $\mathcal{T} = (\Sigma, Q, \Theta)$ be a tiling system with $L = L^{+\omega}(\mathcal{T})$. Let $n = |Q| + 1$ and let p be the $+\omega$-picture of height 1 that corresponds to the ω-word $(b^n a)^\omega$. Since $p \in L$ and $L = L^{+\omega}(\mathcal{T})$, there exists a run r of $\mathcal{T}$ on p. Consider the 1ω-picture $q = (p \times r)$. By our choice of n, there must be two columns $j, j' \in [n]$ with $j < j'$ such that $q_{1j} = q_{1j'}$. Now consider the 1ω-pictures $\tilde{p} = p[1, j] \oplus p[j{+}1, j']^{\oplus\omega}$ and $\tilde{r} = r[1, j] \oplus r[j{+}1, j']^{\oplus\omega}$. It is straightforward to check that $\tilde{r}$ is a run of $\mathcal{T}$ on $\tilde{p}$. Hence, $\tilde{p} \in L^{+\omega}(\mathcal{T})$. However, $\tilde{p}$ does not contain any a and therefore $\tilde{p} \notin L$. A contradiction! □

Büchi-tiling systems and the Büchi-tiling recognizable $+\omega$-picture languages can be viewed as generalizations of Büchi-automata and the ω-regular languages. Recall that a Büchi-automaton $\mathcal{B} = (\Sigma, Q, \Delta, q_0, F)$ consists of the same components as a conventional non-deterministic finite automaton with transition relation $\Delta \subseteq Q \times \Sigma \times Q$. A run r of $\mathcal{B}$ on an ω-word $w \in \Sigma^\omega$ is *accepting* if it visits at least one of the states in F infinitely often. The ω-language *recognized* by $\mathcal{B}$ is the set $L^\omega(\mathcal{B})$ of all ω-words $w \in \Sigma^\omega$ on which $\mathcal{B}$ has an accepting run. We identify ω-words $w \in \Sigma^\omega$ with 1ω-pictures over Σ, and we identify languages $L \subseteq \Sigma^\omega$ of ω-words with $+\omega$-picture languages that contain pictures of height 1 only. It is straightforward to see that for languages of $+\omega$-pictures of height 1, recognizability by Büchi-tiling systems is equivalent to recognizability by Büchi-automata. We prove this fact in the following proposition.

Proposition 2.6. *Let Σ be an alphabet and let $L \subseteq \Sigma^{1\omega}$ (i.e., L is a set of ω-words). The following are equivalent:*

1. *There exists a Büchi-automaton $\mathcal{B}$ such that $L = L(\mathcal{B})$.*
2. *There exists a Büchi-tiling system $\mathcal{S}$ such that $L = L^{+\omega}(\mathcal{S})$.*

Proof. For the direction "(1) ⇒ (2)" let $\mathcal{B} = (\Sigma, Q, \Delta, q_0, F)$ be a Büchi-automaton with $L = L(\mathcal{B})$. We let $\mathcal{S}$ be the Büchi-tiling system with $\mathcal{S} = (\Sigma, Q, \Theta, F)$, where Θ consists of the tiles

#	#
#	(a, q)

and

#	(a, q)
#	#

for all $(q_0, a, q) \in \Delta$, and the tiles

#	#
(a,q)	(b,q')

and

(a,q)	(b,q')
#	#

for all $(q,b,q') \in \Delta$.

It is straightforward to translate a run of the Büchi-automaton $\mathcal{B}$ on a given ω-word $w \in \Sigma^{1\omega}$ into a run of the Büchi-tiling system $\mathcal{S}$ on w, and vice versa, in such a way that the run on $\mathcal{B}$ is accepting iff the according run on $\mathcal{S}$ is accepting.

For the direction "(2) $\Rightarrow$ (1)" let $\mathcal{S} = (\Sigma, Q, \Theta, F)$ be a Büchi-tiling system with $L = L^{+\omega}(\mathcal{S})$. We let $\mathcal{B}$ be the Büchi-automaton with $\mathcal{B} = (\Sigma, Q', q_0, \Delta, F)$, where $Q' = Q \cup \{q_0\}$ for a new state q_0 that does not belong to Q and that will serve as the initial state of $\mathcal{B}$. Furthermore, we let Δ be the transition relation that consists of the following tuples:

• the tuples (q_0, a, q), for all $a \in \Sigma$ and all $q \in Q$ for which Θ contains both tiles

#	#
#	(a,q)

and

#	(a,q)
#	#

• the tuples (q,b,q'), for all $b \in \Sigma$ and all $q, q' \in Q$ for which Θ contains both tiles

#	#
(a,q)	(b,q')

and

(a,q)	(b,q')
#	#

.

It is straightforward to translate a run of the Büchi-tiling system $\mathcal{S}$ on a given ω-word $w \in \Sigma^{1\omega}$ into a run of the Büchi-automaton $\mathcal{B}$ on w, and vice versa, in such a way that the run on $\mathcal{S}$ is accepting iff the according run of $\mathcal{B}$ is accepting. □

It is well-known that Büchi-automata are equivalent to *generalized Büchi-automata*, i.e., Büchi-automata where the acceptance condition F is replaced by an acceptance condition of the form $\{F_1, \ldots, F_k\}$ with $k \in \mathbb{N}_{\geqslant 1}$ and $F_i \subseteq Q$ for every $i \in [k]$. A run r of such an generalized Büchi-automaton on an ω-word w is called *accepting* if for each $i \in [k]$ at least one of the states of F_i occurs infinitely often in r. We use the same generalization for Büchi-tiling systems.

Definition 2.7. *A* generalized Büchi-tiling system *is a 4-tuple* $\mathcal{S} = (\Sigma, Q, \Theta, \tilde{F})$*, where* (Σ, Q, Θ) *is a tiling system, and* $\tilde{F} = \{F_1, \ldots, F_k\}$ *for some* $k \in \mathbb{N}_{\geqslant 1}$ *and sets* $F_1, \ldots, F_k \subseteq Q$*. The set* $\tilde{F}$ *is called the* acceptance condition.

A *run* of $\mathcal{S}$ on a $+\omega$-picture p over Σ is a run of the tiling system $\mathcal{T} = (\Sigma, Q, \Theta)$ on p. A run r is *accepting* if $\inf_1(r) \cap F_i \neq \emptyset$ for *every* $i \in [k]$.

A $+\omega$-picture w over Σ is *accepted* by $\mathcal{S}$ if there exists an accepting run of $\mathcal{S}$ on p. The $+\omega$-picture language *recognized* by $\mathcal{S}$ is the set $L^{+\omega}(\mathcal{S})$ of all $+\omega$-pictures over Σ that are accepted by $\mathcal{S}$.

For translating a generalized Büchi-automaton $\mathcal{B} = (\Sigma, Q, \Delta, q_0, \{F_1, \ldots, F_k\})$ into an equivalent Büchi-automaton $\mathcal{B} = (\Sigma, Q', \Delta', q_0', F)$, one uses the well-known *counting construction*: It suffices to choose $Q' = [k] \times Q$, $q_0' = (1, q_0)$, and $F' = [1] \times F_1$, and to let Δ' be the set consisting of all transitions of the form $\big((i,q), a, (j,q')\big)$, where the following is true: $(q, a, q') \in \Delta$, and $j = i{+}1 \bmod k$ if $q \in F_i$, and $j = i$ otherwise. This construction can easily be adapted to obtain the following.

Proposition 2.8. *Let Σ be an alphabet and let $L \subseteq \Sigma^{+\omega}$. L is Büchi-tiling recognizable if, and only if, there is a generalized Büchi-tiling system $\mathcal{S}$ with $L = L^{+\omega}(\mathcal{S})$.*

Proof. The "only if"-direction is trivial. This is indeed explained by the fact that every Büchi-tiling system $\mathcal{S} = (\Sigma, Q, \Theta, F)$ is equivalent to the generalized Büchi-tiling system $(\Sigma, Q, \Theta, \{F\})$.

For the "if"-direction we use the *counting construction*, i.e., the well-known construction that translates generalized Büchi-automata into equivalent Büchi-automata: When given a state set Q and an acceptance condition $\{F_1, \ldots, F_k\}$, one uses the new state set $Q' = [k] \times Q$. A state $(i, q) \in Q'$ then indicates that we are "waiting to see a state that belongs to F_i". The formal construction is as follows. When given a generalized Büchi-tiling system $\mathcal{S} = (\Sigma, Q, \Theta, \{F_1, \ldots, F_k\})$, we consider the Büchi-tiling system $\mathcal{S}' = (\Sigma, Q', \Theta', F')$ with $Q' = [k] \times Q$ and $F' = [1] \times F_1$ and where Θ' consists of the following tiles:

- for every tile of the form $\begin{array}{|c|c|}\hline \# & \# \\ \hline \# & (a,q) \\ \hline \end{array} \in \Theta$, Θ' contains the tile $\begin{array}{|c|c|}\hline \# & \# \\ \hline \# & \big(a,(1,q)\big) \\ \hline \end{array}$.
- for every tile of the form $\begin{array}{|c|c|}\hline \# & \# \\ \hline (a,q) & (b,p) \\ \hline \end{array} \in \Theta$ and every $i \in [k]$, Θ' contains the tile $\begin{array}{|c|c|}\hline \# & \# \\ \hline \big(a,(i,q)\big) & \big(b,(j,p)\big) \\ \hline \end{array}$ where $j \equiv i+1 \bmod k$ if $q \in F_i$, and $j = i$ otherwise.
- for every tile of the form $\begin{array}{|c|c|}\hline \# & (a,q) \\ \hline \# & (b,p) \\ \hline \end{array} \in \Theta$, Θ' contains the tile

$\begin{array}{|c|c|}\hline \# & (a,(1,q)) \\ \hline \# & (b,(1,p)) \\ \hline \end{array}$.

• for every tile of the form $\begin{array}{|c|c|}\hline (a,q) & (b,p) \\ \hline (a',q') & (b',p') \\ \hline \end{array} \in \Theta$ and all $i, i' \in [k]$, Θ' contains the tile $\begin{array}{|c|c|}\hline (a,(i,q)) & (b,(j,p)) \\ \hline (a',(i',q')) & (b',(j',p')) \\ \hline \end{array}$, where $j \equiv i+1 \bmod k$ if $q \in F_i$, and $j = i$ otherwise, and $j' \equiv i'+1 \bmod k$ if $q' \in F_{i'}$, and $j' = i'$ otherwise.

• for every tile of the form $\begin{array}{|c|c|}\hline \# & (a,q) \\ \hline \# & \# \\ \hline \end{array} \in \Theta$, Θ' contains the tile $\begin{array}{|c|c|}\hline \# & (a,(1,q)) \\ \hline \# & \# \\ \hline \end{array}$.

• for every tile of the form $\begin{array}{|c|c|}\hline (a,q) & (b,p) \\ \hline \# & \# \\ \hline \end{array} \in \Theta$ and every $i \in [k]$, Θ' contains the tile $\begin{array}{|c|c|}\hline (a,(i,q)) & (b,(j,p)) \\ \hline \# & \# \\ \hline \end{array}$, where $j \equiv i+1 \bmod k$ if $q \in F_i$, and $j = i$ otherwise.

It is straightforward to translate a run of $\mathcal{S}$ on a given $+\omega$-picture $p \in \Sigma^{+\omega}$, into a run of $\mathcal{S}'$ on p, and vice versa, in such a way that the run of $\mathcal{S}'$ is accepting iff the according run of $\mathcal{S}$ is accepting. □

2.3 Closure Properties of Recognizable $+\omega$-Picture Languages

Our goal in this section is to show that the class of Büchi-tiling recognizable $+\omega$-picture languages is closed under the operations projection, union, and intersection, but not under complementation. Techniques known for tiling systems over finite pictures (see [63, 64]) can be transferred to our setting. In addition, we provide a characterization that relates the Büchi-tiling recognizable $+\omega$-picture languages to tiling recognizable languages of finite pictures.

Proposition 2.9. *Let Σ be an alphabet of size $|\Sigma| \geqslant 2$. The family of Büchi-tiling recognizable $+\omega$-picture languages over Σ is closed under projection, union, and intersection.*

Proof. Closure under projection. Let $\mathcal{S} = (\Sigma, Q, \Theta, F)$ be a Büchi-tiling system and let $L = L^{+\omega}(\mathcal{S})$ be the $+\omega$-picture language recognized by $\mathcal{S}$. Let Σ' be an alphabet, let $\pi : \Sigma \to \Sigma'$ be a projection, and let $L' = \pi(L)$.

Our goal is to construct a Büchi-tiling system $\mathcal{S}'$ that recognizes L'. To this end, let $\mathcal{S}' = (\Sigma', Q', \Theta', F')$ be the Büchi-tiling system with $Q' = \Sigma \times Q$ and $F' = \Sigma \times F$, where Θ' is obtained as follows: Consider all tiles

$$\begin{array}{|c|c|}\hline t_{11} & t_{12} \\ \hline t_{21} & t_{22} \\ \hline \end{array} \in \Theta$$

and add to Θ' only all the tiles of the form

$$\begin{array}{|c|c|}\hline v_{11} & v_{12} \\ \hline v_{21} & v_{22} \\ \hline \end{array}$$

where for all $i, j \in \{1, 2\}$ we have

$$v_{ij} = \begin{cases} \# & \text{if } \ t_{ij} = \#, \\ \big(\, \pi(a_{ij}),\ (a_{ij}, q_{ij})\,\big) & \text{if } t_{ij} = (a_{ij}, q_{ij}). \end{cases}$$

It is straightforward to verify that a $+\omega$-picture p' over Σ' is accepted by $\mathcal{S}'$ if, and only if, there is a $+\omega$-picture p over Σ such that $p' = \pi(p)$ and p is accepted by $\mathcal{S}$. Hence, $L^{+\omega}(\mathcal{S}') = L' = \pi(L)$, and the class of Büchi-tiling recognizable $+\omega$-picture languages is closed unter projection.

Closure under union and intersection. For $i \in \{1, 2\}$ let $\mathcal{S}_i = (\Sigma, Q_i, \Theta_i, F_i)$ be a Büchi-tiling system and let $L_i = L^{+\omega}(\mathcal{S}_i)$ be the $+\omega$-picture language recognized by $\mathcal{S}_i$. Without loss of generality we can assume that $Q_1 \cap Q_2 = \emptyset$.

Let $\mathcal{S}_\cup = (\Sigma, Q_\cup, \Theta_\cup, F_\cup)$ be the Büchi-tiling system with $Q_\cup = Q_1 \cup Q_2$, $\Theta_\cup = \Theta_1 \cup \Theta_2$, and $F_\cup = F_1 \cup F_2$. It is straightforward to verify that a $+\omega$-picture p over Σ is accepted by $\mathcal{S}_\cup$ if, and only if, it is accepted by $\mathcal{S}_1$ or by $\mathcal{S}_2$. Hence, $L^{+\omega}(\mathcal{S}_\cup) = L_1 \cup L_2$, and the class of Büchi-tiling recognizable $+\omega$-picture languages is closed under union.

Let $\mathcal{S}_\cap = (\Sigma, Q_\cap, \Theta_\cap, \tilde{F})$ be the *generalized* Büchi-tiling system with $Q_\cap = Q_1 \times Q_2$, $\tilde{F} = \{F_1', F_2'\}$ with $F_1' = F_1 \times Q_2$ and $F_2' = Q_1 \times F_2$, and where $\Theta_\cap$ is obtained as follows: Consider all pairs of tiles

$$\begin{array}{|c|c|}\hline s_{11} & s_{12} \\ \hline s_{21} & s_{22} \\ \hline \end{array} \in \Theta_1 \quad \text{and} \quad \begin{array}{|c|c|}\hline t_{11} & t_{12} \\ \hline t_{21} & t_{22} \\ \hline \end{array} \in \Theta_2,$$

where the following is true for all $i, j \in \{1, 2\}$: Either $s_{ij} = t_{ij} = \#$ or there exists a letter $a_{ij} \in \Sigma$ and states $q'_{ij} \in Q_1$ and $q''_{ij} \in Q_2$ such that

$s_{ij} = (a_{ij}, q'_{ij})$ and $t_{ij} = (a_{ij}, q''_{ij})$. Then add to $\Theta_\cap$ all tiles of the form

v_{11}	v_{12}
v_{21}	v_{22}

where for all $i, j \in \{1, 2\}$ we let

$$v_{ij} = \begin{cases} \# & \text{if } s_{ij} = t_{ij} = \#, \\ \big(a_{ij},\, (q'_{ij}, q''_{ij})\big) & \text{otherwise.} \end{cases}$$

It is straightforward to verify that a $+\omega$-picture p over Σ is accepted by $\mathcal{S}_\cap$ if, and only if, it is accepted by $\mathcal{S}_1$ and by $\mathcal{S}_2$. Thus, $L^{+\omega}(\mathcal{S}_\cap) = L_1 \cap L_2$. Applying Proposition 2.8 we obtain that $L_1 \cap L_2$ is Büchi-tiling recognizable. Hence, the class of Büchi-tiling recognizable $+\omega$-picture languages is closed under intersection. □

Lemma 2.10. *Let Σ be an alphabet and let $L \subseteq \Sigma^{++}$ be a tiling recognizable language of finite pictures over Σ. Then, the following is true:*

1. *For every Büchi-tiling recognizable $+\omega$-picture language $L' \subseteq \Sigma^{+\omega}$, the $+\omega$-picture language $L \oplus L'$ is Büchi-tiling recognizable.*
2. *The $+\omega$-picture language $L^{\oplus\omega}$ is Büchi-tiling recognizable.*

Proof. 1. Let Σ be an alphabet, let $L_1 \subseteq \Sigma^{++}$ be a language of finite pictures that is recognized by a tiling system $\mathcal{T}_1 = (\Sigma, Q_1, \Theta_1)$, and let $L_2 \subseteq \Sigma^{+\omega}$ be a language of $+\omega$-pictures that is recognized by a Büchi-tiling system $\mathcal{S}_2 = (\Sigma, Q_2, \Theta_2, F_2)$. Without loss of generality we can assume that $Q_1 \cap Q_2 = \emptyset$. Our goal is to construct a Büchi-tiling system $\mathcal{S} = (\Sigma, Q, \Theta, F)$ that recognizes the $+\omega$-language $L = L_1 \oplus L_2$. We choose

$$Q \;=\; Q_1 \cup Q_2 \qquad \text{and} \qquad F = F_2\,.$$

The idea of the construction for Θ is as follows: Let p be a $+\omega$-picture of the form $p_1 \oplus p_2$ with $p_1 \in L_1$ and $p_2 \in L_2$. Let r_1 be a successful run of $\mathcal{T}_1$ on p_1, and let r_2 be a successful run of $\mathcal{S}_2$ on p_2. We will construct Θ in such a way that $r_1 \oplus r_2$ is a successful run of $\mathcal{S}$ on p. The set of tiles Θ is built as follows:

Let Θ'_1 be the set of tiles obtained from Θ_1 by removing all tiles that have #-symbols in both entries of their second column, and let Θ'_2 be the set of tiles obtained from Θ_2 by removing all tiles that have #-symbols in both entries of their first column. Then add to Θ the set of tiles $\Theta'_1 \cup \Theta'_2$.

• Consider all pairs of tiles $\begin{array}{|c|c|}\hline \# & \# \\ \hline (a,q) & \# \\ \hline \end{array} \in \Theta_1$ and $\begin{array}{|c|c|}\hline \# & \# \\ \hline \# & (b,p) \\ \hline \end{array} \in \Theta_2$ and add to Θ the tile $\begin{array}{|c|c|}\hline \# & \# \\ \hline (a,q) & (b,p) \\ \hline \end{array}$.

• Consider all pairs of tiles $\begin{array}{|c|c|}\hline (a,q) & \# \\ \hline (a',q') & \# \\ \hline \end{array} \in \Theta_1$ and $\begin{array}{|c|c|}\hline \# & (b,p) \\ \hline \# & (b',p') \\ \hline \end{array} \in \Theta_2$ and add to Θ the tile $\begin{array}{|c|c|}\hline (a,q) & (b,p) \\ \hline (a',q') & (b',p') \\ \hline \end{array}$.

• Consider all pairs of tiles $\begin{array}{|c|c|}\hline (a,q) & \# \\ \hline \# & \# \\ \hline \end{array} \in \Theta_1$ and $\begin{array}{|c|c|}\hline \# & (b,p) \\ \hline \# & \# \\ \hline \end{array} \in \Theta_2$ and add to Θ the tile $\begin{array}{|c|c|}\hline (a,q) & (b,p) \\ \hline \# & \# \\ \hline \end{array}$.

It is straightforward to verify that a $+\omega$-picture p over Σ is accepted by $\mathcal{S}$ if, and only if, p is of the form $p_1 \oplus p_2$ for a finite picture p_1 that is accepted by $\mathcal{T}_1$ and a $+\omega$-picture p_2 that is accepted by $\mathcal{S}_2$.

2. Let $\mathcal{T} = (\Sigma, Q, \Theta)$ be a tiling system with $L = L^{++}(\mathcal{T})$. Our aim is to construct a Büchi-tiling system $\mathcal{S} = (\Sigma, Q', \Theta', F')$ with $L^{+\omega}(\mathcal{S}) = L^{\oplus\omega}$. Let $\smile$, $|$ and be two distinct symbols. We choose

$$Q' \;=\; Q \times \{\smile, |\} \qquad \text{and} \qquad F' \;=\; Q \times \{|\}\,.$$

The idea for constructing Θ' is as follows: Consider a $+\omega$-picture $p \in L^{\oplus\omega}$ of the form $p_1 \oplus p_2 \oplus p_3 \oplus \cdots$ where $p_\nu \in L^{++}(\mathcal{T})$ for all $\nu \in \mathbb{N}_{\geqslant 1}$. For every $\nu \in \mathbb{N}_{\geqslant 1}$ let r_ν be a run of $\mathcal{T}$ on p_ν (in particular, r_ν is a finite picture over Q of the same size as p_ν). Let r'_ν be the picture obtained from r_ν as follows: For every row i and every column j of r_ν, consider the entry q of r_ν in row i and column j. If j is the *rightmost* column of r_ν, replace q by $(q, |)$; otherwise replace q by $(q, \smile)$. We will construct Θ' in such a way that the $+\omega$-picture $r' = r'_1 \oplus r'_2 \oplus r'_3 \oplus \cdots$ is a run of $\mathcal{S}'$ on p. The set Θ' is defined as follows:

• Consider all tiles of the form $\begin{array}{|c|c|}\hline \# & \# \\ \hline \# & (a,q) \\ \hline \end{array}$, $\begin{array}{|c|c|}\hline \# & (a,q) \\ \hline \# & (a',q') \\ \hline \end{array}$, $\begin{array}{|c|c|}\hline \# & (a,q) \\ \hline \# & \# \\ \hline \end{array} \in \Theta$ with $(a,q), (a',q') \in \Sigma \times Q$. Then for all $x \in \{\smile, |\}$, we add to Θ' the following tiles, respectively:

$$\begin{array}{|c|c|}\hline \# & \# \\ \hline \# & (a,(q,x)) \\ \hline \end{array}\,, \quad \begin{array}{|c|c|}\hline \# & (a,(q,x)) \\ \hline \# & (a',(q',x)) \\ \hline \end{array}\,, \quad \begin{array}{|c|c|}\hline \# & (a,(q,x)) \\ \hline \# & \# \\ \hline \end{array}\,.$$

• Consider all tiles of the form $\begin{array}{|c|c|}\hline \# & \# \\ \hline (b,p) & (a,q) \\ \hline \end{array}$, $\begin{array}{|c|c|}\hline (b,p) & (a,q) \\ \hline \# & \# \\ \hline \end{array} \in \Theta$ with $(b,p),(a,q) \in \Sigma \times Q$. Then for all $x \in \{\smile, |\}$, we add to Θ' the following tiles, respectively:

$$\begin{array}{|c|c|}\hline \# & \# \\ \hline (b,(p,\smile)) & (a,(q,x)) \\ \hline \end{array}, \begin{array}{|c|c|}\hline (b,(p,\smile)) & (a,(q,x)) \\ \hline \# & \# \\ \hline \end{array}.$$

• Consider all tiles of the form $\begin{array}{|c|c|}\hline (b,p) & (a,q) \\ \hline (b',p') & (a',q') \\ \hline \end{array} \in \Theta$ with $(b,p),(b',p'),(a,q),(a',q') \in \Sigma \times Q$. Then for all $x \in \{\smile, |\}$, we add to Θ' the following tiles:

$$\begin{array}{|c|c|}\hline (b,(p,\smile)) & (a,(q,x)) \\ \hline (b',(p',\smile)) & (a',(q',x)) \\ \hline \end{array}.$$

• Consider all pairs of tiles in Θ of the form

$$\begin{array}{|c|c|}\hline \# & \# \\ \hline (b,p) & \# \\ \hline \end{array} \quad \text{and} \quad \begin{array}{|c|c|}\hline \# & \# \\ \hline \# & (a,q) \\ \hline \end{array}$$

with $(b,p),(a,q) \in \Sigma \times Q$. Then for all $x \in \{\smile, |\}$, we add to Θ' the tiles

$$\begin{array}{|c|c|}\hline \# & \# \\ \hline (b,(p,|)) & (a,(q,x)) \\ \hline \end{array}.$$

• Consider all pairs of tiles in Θ of the form

$$\begin{array}{|c|c|}\hline (b,p) & \# \\ \hline (b',p') & \# \\ \hline \end{array} \quad \text{and} \quad \begin{array}{|c|c|}\hline \# & (a,q) \\ \hline \# & (a',q') \\ \hline \end{array}$$

with $(b,p),(b',p'),(a,q),(a',q') \in \Sigma \times Q$. Then for all $x \in \{\smile, |\}$, we add to Θ' the tiles

$$\begin{array}{|c|c|}\hline (b,(p,|)) & (a,(q,x)) \\ \hline (b',(p',|)) & (a',(q',x)) \\ \hline \end{array}.$$

• Consider all pairs of tiles in Θ of the form

$$\begin{array}{|c|c|}\hline (b,p) & \# \\ \hline \# & \# \\ \hline \end{array} \quad \text{and} \quad \begin{array}{|c|c|}\hline \# & (a,q) \\ \hline \# & \# \\ \hline \end{array}$$

with $(b,p),(a,q) \in \Sigma \times Q$. Then for all $x \in \{\smile, |\}$, we add to Θ' the tiles

$$\begin{array}{|c|c|}\hline (b,(p,|)) & (a,(q,x)) \\ \hline \# & \# \\ \hline \end{array}.$$

It is straightforward (but tedious) to verify $L^{+\omega}(\mathcal{S}) = L^{\textcircled{D}\omega}$.

□

Proposition 2.11. *Let Σ be an alphabet of size $|\Sigma| \geqslant 2$. The family of Büchi-tiling recognizable $+\omega$-picture languages over Σ is not closed under complement.*

Proof. Let Σ be an alphabet with $|\Sigma| \geqslant 2$. Let $L_1 \subseteq \Sigma^{++}$ be the language of all finite pictures of the form $s \oplus s$ for all $s \in \bigcup_{m \in \mathbb{N}_{\geqslant 1}} \Sigma^{m,m}$. We claim that the $+\omega$-picture language $L = L_1 \oplus \Sigma^{+\omega}$ is *not* Büchi-tiling recognizable, while its complement $\Sigma^{+\omega} \setminus L$ *is* Büchi-tiling recognizable. To prove this claim, we use results and techniqes from [64]. From [64] we know the following:

(i) The picture language L_1 is *not* tiling recognizable.

(ii) The picture language $\Sigma^{++} \setminus L_1$ is tiling-recognizable.

(iii) The following picture language is tiling recognizable:

$$L_1' \;=\; \{\, s \oplus s' \;:\; s, s' \in \Sigma^{++},\ \ell_1(s) = \ell_1(s') = \ell_2(s) = \ell_2(s'),\ s \neq s' \,\}.$$

It is easy to see that $\Sigma^{+\omega} \setminus L \;=\; L_1' \oplus \Sigma^{+\omega}$. Clearly, $\Sigma^{+\omega}$ is Büchi-tiling recognizable. Using (iii) and Lemma 2.10, we obtain that the $+\omega$-picture language $L_1' \oplus \Sigma^{+\omega} \;=\; \Sigma^{+\omega} \setminus L$ is Büchi-tiling recognizable. For contradiction, let us assume now that L is Büchi-tiling recognizable, i.e., $L = L^{+\omega}(\mathcal{S})$ for a Büchi-tiling system $\mathcal{S} = (\Sigma, Q, \Delta, F)$. We adapt the counting argument used in [64] for proving (i) to our setting. For the remainder of this proof, let us fix a number $m \in \mathbb{N}_{\geqslant 1}$ such that

$$|\Sigma|^{(m^2)} \;>\; \big(|\Sigma \times Q|\big)^{2m} \tag{2.1}$$

(note that for fixed finite non-empty sets Σ and Q this inequality is true for all sufficiently large m, since the function $2^{(x^2)}$ grows faster than the function 2^{cx} for any constant c). Furthermore, let us fix a letter $a \in \Sigma$. Let a_m^ω be the $+\omega$-picture of height m that has letter a in *all* rows and columns. Furthermore, for every finite picture $s \in \Sigma^{m,m}$ (i.e., s is a square of height m and width m) let p_s be the $+\omega$-picture of height m defined via $p_s = s \oplus s \oplus a_m^\omega$. Clearly, $p_s \in L$. By our assumption $L = L^{+\omega}(\mathcal{S})$, there exists an accepting run r_s of $\mathcal{S}$ on p_s. Let $q_s = (p_s \times r_s)$. Now, consider the stripes $q_s[m, m{+}1]$ consisting of the m-th and the $(m{+}1)$-st column of q_s, for all $s \in \Sigma^{m,m}$. Due to the inequality (2.1), there must exist two distinct $s, s' \in \Sigma^{m,m}$ such that the stripes $q_s[m, m{+}1]$ and $q_{s'}[m, m{+}1]$ are identical, i.e., $q_s[m, m{+}1] = q_{s'}[m, m{+}1]$. We choose $\tilde{r}_s$ to be the $+\omega$-picture over Q that is obtained from r_s by replacing the first m columns of r_s with the first m columns of $r_{s'}$. It is straightforward to check that $\tilde{r}_s$ is an accepting run of $\mathcal{S}$ on the $+\omega$-picture $s' \oplus s \oplus a_m^\omega$. This $+\omega$-picture, however, does *not* belong to L. This contradicts our assumption that $L^{+\omega}(\mathcal{S}) = L$. $\square$

It is well-known (see e.g., [97]) that the ω-regular word-languages are exactly the languages of ω-words that are unions of finitely many ω-languages of the form $L_1 \cdot L_2^\omega$, where L_1, L_2 are regular languages of finite words. It is tempting to conjecture that the same holds true for Büchi-tiling recognizable languages of $+\omega$-pictures. Indeed, by Lemma 2.10 and Proposition 2.9 we obtain the "easy direction" of the characterization theorem: If L is a $+\omega$-picture language that is the union of a finite number of sets of the form $L_1 \oplus L_2^{\oplus\omega}$, where $L_1, L_2 \subseteq \Sigma^{++}$ are tiling recognizable sets of finite pictures, then L is Büchi-tiling recognizable. The opposite direction, however, is not true — even if we drop the requirement that L_1 and L_2 are tiling recognizable. We will prove this later in Section 2.6 using combinatorial arguments.

2.4 A Logical Characterization of the Büchi-Tiling Recognizable $+\omega$-Picture Languages

In Section 1.4, we briefly recalled some notions and ideas of picture languages in connection with existential monadic second-order logic (EMSO). Here, we provide more details on connection between picture languages and logic. As the class of recognizable picture languages is not closed under complementation [63, 67, 68, 69], but MSO is closed under negation, a characterization of the recognizable picture languages by MSO is not conceivable. A characterization by EMSO, however, has been obtained in [64]: the recognizable picture languages are exactly the picture languages that are definable in EMSO. For this, the authors of [64] use the signature $\tau_\Sigma = \{S_v, S_h\} \cup \{P_a : a \in \Sigma\}$ which consists of two binary relation symbols S_v and S_h and a unary relation symbol P_a for every letter $a \in \Sigma$. A finite picture p of size (m, n) over Σ is represented by a finite relational structure $\underline{p} = \big(\mathrm{dom}(p),\ S_v^p,\ S_h^p,\ (P_a^p)_{a\in\Sigma}\big)$, of signature τ_Σ where,

- the domain $\mathrm{dom}(p) = [m] \times [n]$ consists of all *positions* or *pixels* of p,
- for every $a \in \Sigma$, the relation P_a^p consists of all positions $(i, j) \in \mathrm{dom}(p)$ with $p_{i,j} = a$,
- S_v^p is the vertical successor relation on the positions of p,
- S_h^p is the horizontal successor relation on the positions of p.

We use the same representation for $+\omega$-pictures p, where the domain $\mathrm{dom}(p)$ of a $+\omega$-picture of height m is defined as $\mathrm{dom}(p) := [m] \times \mathbb{N}_{\geqslant 1}$.

As usual, we will use a countable set $\mathcal{V}_i$ of *first-order variables* and a countable set $\mathcal{V}_s$ of *set* or *second-order variables* to describe picture languages by logical formulas . The letters like $x, y, z, x_1, x_2, \ldots$ are used to denote first-order variables, and the letters like $X, Y, Z, X_1, X_2, \ldots$ are used to denote second-order variables. The set $\mathrm{FO}[\tau_\Sigma]$ of all *first-order formulas* of signature τ_Σ is inductively defined as follows:

- $\mathrm{FO}[\tau_\Sigma]$ contains all *atomic formulas* of the form $x{=}y$, $P_a(x)$, $x \in X$, xS_vy, and xS_hy, for all first-order variables $x, y \in \mathcal{V}_i$, all letters $a \in \Sigma$, and all second-order variables $X \in \mathcal{V}_s$.
- $\mathrm{FO}[\tau_\Sigma]$ is closed under Boolean combinations, i.e., whenever φ and ψ belong to $\mathrm{FO}[\tau_\Sigma]$, then $\mathrm{FO}[\tau_\Sigma]$ also contains the formulas $\neg\varphi$, $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, $(\varphi \to \psi)$, and $(\varphi \leftrightarrow \psi)$.
- $\mathrm{FO}[\tau_\Sigma]$ is closed under existential and universal quantification of first-order variables, i.e., whenever φ belongs to $\mathrm{FO}[\tau_\Sigma]$ and $x \in \mathcal{V}_i$, then $\mathrm{FO}[\tau_\Sigma]$ also contains the formulas $\exists x\, \varphi$ and $\forall x\, \varphi$.

Note that we already explained the intended meaning of these formulas in Section 1.4. Now for $\varphi \in \mathrm{FO}[\tau_\Sigma]$, the set $\mathit{free}(\varphi)$ of all *free variables* of φ consists of all second-order variables occurring in φ and all first-order variables x that have at least one free occurrence in φ, i.e., an occurrence that is not within the range of a quantifier of the form $\exists x$ or $\forall x$. If $\mathit{free}(\varphi) \subseteq \{X_1, \ldots, X_k, x_1, \ldots, x_\ell\}$, (finite or $+\omega$-)picture over Σ, $A_1, \ldots, A_k$ are subsets of $\mathrm{dom}(p)$, and $a_1, \ldots, a_\ell$ are elements in $\mathrm{dom}(p)$, we write $\big(\underline{p}, A_1, \ldots, A_k, a_1, \ldots, a_\ell\big) \models \varphi$ to indicate that the statement made by φ is true in $\underline{p}$ when assigning the second-order variable X_i to the set A_i and assigning the fisrt-order variable x_j to the position a_j, for all $i \in [k]$, $j \in [\ell]$. We often abbreviate sequences $A_1, \ldots, A_k$ and $a_1, \ldots, a_\ell$ by $\overline{A}$ and $\overline{a}$. The set $\mathrm{EMSO}[\tau_\Sigma]$ of *existential monadic second-order formulas* of the signature τ_Σ consists of all formulas Φ of the form $\exists X_1 \cdots \exists X_k\, \varphi$, where $k \geqslant 0$ and $\varphi \in \mathrm{FO}[\tau_\Sigma]$. The set of all free variables of Φ is defined as $\mathit{free}(\Phi) = \mathit{free}(\varphi) \setminus \{X_1, \ldots, X_k\}$. If $\mathit{free}(\Phi) \subseteq \{X_{k+1}, \ldots, X_{k+k'}, x_1, \ldots, x_\ell\}$, p is a picture over Σ, $\overline{A} = A_{k+1}, \ldots, A_{k+k'} \subseteq \mathrm{dom}(p)$, and $\overline{a} = a_1, \ldots, a_\ell \in \mathrm{dom}(p)$, then $(\underline{p}, \overline{A}, \overline{a})$ *satisfies* Φ, denoted by $(\underline{p}, \overline{A}, \overline{a}) \models \Phi$, if there exist sets $A_1, \ldots, A_k \subseteq \mathrm{dom}(p)$ such that $(\underline{p}, A_1, \ldots, A_k, \overline{A}, \overline{a}) \models \varphi$. Recall that *sentences* are formulas Φ with $\mathit{free}(\Phi) = \emptyset$. For a *sentence* Φ we simply write $\underline{p} \models \Phi$.

Definition 2.12. *Let Σ be an alphabet and let $\bullet$ be one of the symbols $+$ or ω. The $+\bullet$-picture language* defined *by an* $\mathrm{EMSO}[\tau_\Sigma]$*-sentence Φ is the*

set $L^{+\bullet}(\Phi) := \{\ p \in \Sigma^{+\bullet} : \underline{p} \models \Phi\ \}$. *Let* $\mathcal{L} \subseteq \mathrm{EMSO}[\tau_\Sigma]$. *A* $+\bullet$*-picture language* $L \subseteq \Sigma^{+\bullet}$ *is* $\mathcal{L}$-definable *if there exists a sentence* $\Phi \in \mathcal{L}$ *such that* $L = L^{+\bullet}(\Phi)$.

Giammarresi et al. [64] have shown that a language of *finite* pictures is tiling recognizable if, and only if, it is $\mathrm{EMSO}[\tau_\Sigma]$-definable. Their characterization does *not* carry over to languages of $+\omega$-pictures. See the following proposition:

Proposition 2.13. *Let* $\Sigma = \{a, b\}$ *consist of two distinct letters. Let* $L = \{p \in \Sigma^{1\omega} : p$ *contains at least one occurrence of the letter* $a\}$*, and let* $L' = \{p \in \Sigma^{1\omega} : p$ *contains infinitely many* a*'s*$\}$*. The following holds:*

1. L *is* $\mathrm{FO}[\tau_\Sigma]$*-definable, but not tiling recognizable.*

2. L' *is Büchi-tiling recognizable, but not* $\mathrm{EMSO}[\tau_\Sigma]$*-definable.*

We will give the complete proof of this proposition at the end of Section 2.5, when we have introduced all the required notions and tools.

To obtain a logical characterization of the Büchi-tiling recognizable $+\omega$-picture languages, we extend EMSO by quantifiers of the form $\exists^\infty X$, for set variables $X \in \mathcal{V}_s$, with the intended meaning that *there exists an* ***infinite*** *set* X. We write $\mathrm{EMSO}^\infty[\tau_\Sigma]$ for the set of all formulas Ψ of the form

$$\exists^\infty X_1 \cdots \exists^\infty X_k\, \Phi$$

where $k \geqslant 0$, $X_1, \ldots, X_k \in \mathcal{V}_s$, and $\Phi \in \mathrm{EMSO}[\tau_\Sigma]$. The set of free variables of Ψ is $\mathit{free}(\Psi) = \mathit{free}(\Phi) \setminus \{X_1, \ldots, X_k\}$. If $\mathit{free}(\Psi) \subseteq \{X_{k+1}, \ldots, X_{k+k'}, x_1, \ldots, x_\ell\}$, p is a $+\omega$-picture over Σ, $\overline{A} = A_{k+1}, \ldots, A_{k+k'} \subseteq \mathrm{dom}(p)$, and $\overline{a} = a_1, \ldots, a_\ell \in \mathrm{dom}(p)$, then $(\underline{p}, \overline{A}, \overline{a})$ *satisfies* Ψ (in symbols: $(\underline{p}, \overline{A}, \overline{a}) \models \Psi$) if there exist *infinite* sets $A_1, \ldots, A_k \subseteq \mathrm{dom}(p)$ such that $(\underline{p}, A_1, \ldots, A_k, \overline{A}, \overline{a}) \models \Phi$.

It is not difficult to see that $\mathrm{EMSO}^\infty[\tau_\Sigma]$ is expressive enough to describe all Büchi-tiling recognizable $+\omega$-picture languages. For the opposite direction, we apply a modified approach of Giammarresi et al. [64]. The main step is to translate a given $\mathrm{FO}[\tau_\Sigma]$-formula $\varphi(X_1, \ldots, X_k)$, with free set variables $X_1, \ldots, X_k$, into a generalized Büchi-tiling system over the extended alphabet $\Sigma \times \{0, 1\}^k$; note that a position that carries a letter $(a, (\alpha_1, \ldots, \alpha_k))$ of this extended alphabet corresponds to a position that carries the letter $a \in \Sigma$ and, for each $i \in [k]$, belongs to the set X_i iff $\alpha_i = 1$.

Afterwards, we lift the translation so that it applies also to $\mathrm{EMSO}^{\infty}[\tau_{\Sigma}]$-sentences. Due to the equivalence of generalized Büchi-tiling systems and Büchi-tiling systems, we then obtain the following:

Theorem 2.14. *Let Σ be an alphabet and let $L \subseteq \Sigma^{+\omega}$. L is Büchi-tiling recognizable if, and only if, L is* $\mathrm{EMSO}^{\infty}[\tau_{\Sigma}]$*-definable.*

The next section is devoted to the proof of this result.

Remark 2.15. *Note that some other extensions of* EMSO *by an infinity quantifier has been considered in the literature. For example the extension of* EMSO *by an additional infinity first-order quantifier $\exists^{\infty} x\varphi$ (see [20]) which means for infinitely many position x, the statement made by formula φ holds. Here, we have our additional infinity quantifier over second-order variables.*

2.5 Definability Equals Recognizability

To start with the steps of the proof of Theorem 2.14, it will be convenient to introduce the following $\mathrm{FO}[\tau_{\Sigma}]$-formulas:

- $\varphi_t(x) = \neg\, \exists y\, y S_h x$, which expresses that x is a position in the top row of a picture.
- $\varphi_b(x) = \neg\, \exists y\, x S_h y$, which expresses that x is a position in the bottom row of a picture.
- $\varphi_l(x) = \neg\, \exists y\, y S_v x$, which expresses that x is a position in the leftmost column of a picture.
- $\varphi_{tl}(x) = \varphi_t(x) \wedge \varphi_l(x)$, which expresses that x is the upper left corner of a picture.
- $\varphi_{bl}(x) = \varphi_b(x) \wedge \varphi_l(x)$, which expresses that x is the bottom left corner of a picture.

Now we first show that recognizability implies definability:

Theorem 2.16. *Let Σ be an alphabet and let $L \subseteq \Sigma^{+\omega}$ be a Büchi-tiling recognizable $+\omega$-picture language. Then L is* $\mathrm{EMSO}^{\infty}[\tau_{\Sigma}]$*-definable.*

Proof. The proof is based on the standard approach for other variants of the Büchi-Elgot-Trakhtenbrot theorem (cf., e.g., [97]): Let $L \subseteq \Sigma^{+\omega}$ be Büchi-tiling recognizable, and let $\mathcal{S} = (\Sigma, Q, \Theta, F)$ be a Büchi-tiling system

such that $L^{+\omega}(\mathcal{S}) = L$. Without loss of generality, we can assume that $Q = [k]$ for some $k \in \mathbb{N}_{\geqslant 1}$. By assumption we know for every $+\omega$-picture p over Σ that $p \in L$ if, and only if, there is a $+\omega$-picture r over Q of the same height as p such that the $+\omega$-picture $(p \times r)$ satisfies $T_{2,2}(\widehat{p \times r}) \subseteq \Theta$. We formalize the latter be an $\mathrm{EMSO}^{\infty}[\tau_\Sigma]$-sentence Φ of the form

$$\exists^{\infty} Z \; \exists X_1 \cdots \exists X_k \; \varphi \,.$$

The intended meaning is that for every $q \in Q$, the set variable X_q is interpreted by the set of all positions of a given picture $p \in \Sigma^{+\omega}$ that are assigned to the state q by a run of $\mathcal{S}$ on p. Furthermore, we have to express that a state from F appears infinitely often in the run of $\mathcal{S}$ on p. For this, the set variable Z under the quantifier $\exists^{\infty}$ is interpreted by a subset of $\bigcup_{q \in F} X_q$ which contains only positions that belong to the first row. Then, the $\mathrm{FO}[\tau_\Sigma]$-formula φ states that

(1) Every position of p belongs to exactly one of the sets $X_1, \ldots, X_k$. We can thus identify an assignment to $X_1, \ldots, X_k$ with the mapping $r : \mathrm{dom}(p) \to Q$ that maps every position x of p to the (unique) state $q \in Q$ with $x \in X_q$.

(2) Every subpicture of $\widehat{p \times r}$ of size $(2, 2)$ belongs to Θ.

(3) There is an infinite set of positions which contains only positions in the first row of p and are labelled with a state from F.

All this can be formulated in the following sentence:

$$\begin{aligned}
\Phi := \exists^{\infty} Z \; \exists X_1 \; ... \; \exists X_k \; \Big[\; & \forall x \bigvee_{1 \leqslant i \leqslant k} \Big(x \in X_i \; \wedge \bigwedge_{i \neq j} \neg (x \in X_j) \Big) \\
& \wedge \forall \, x_1, x_2, x_3, x_4 \; (x_1 S_h x_2 \wedge x_3 S_h x_4 \wedge x_1 S_v x_3 \wedge x_2 S_v x_4) \; \rightarrow \\
& \wedge \bigvee_{\begin{array}{|c|c|}\hline (a_1, q_1) & (a_2, q_2) \\ \hline (a_3, q_3) & (a_4, q_4) \\ \hline \end{array} \in \Theta} \; \bigwedge_{1 \leqslant i \leqslant 4} P_{a_i}(x_i) \; \wedge \; x_i \in X_{q_i} \\
& \wedge \forall x \, \Big(x \in Z \; \rightarrow \; \big(\varphi_t(x) \; \wedge \bigvee_{q \in F} x \in X_q \; \big) \Big) \Big].
\end{aligned}$$

□

The rest of this section is devoted to the proof of the "if"-direction of Theorem 2.14. Namely the following theorem:

Theorem 2.17. *Let Σ be an alphabet and let $L \subseteq \Sigma^{+\omega}$ be* $\mathrm{EMSO}^{\infty}[\tau_\Sigma]$*-definable. Then L is Büchi-tiling recognizable $+\omega$-picture language.*

To prove this theorem, we follow the overall approach of Giammarresi et al. [64] with some modifications. The first step taken in [64] is to show that a picture language is *first-order definable* if and only if it is *locally threshold testable*. The same result holds true for $+\omega$-picture languages. For giving a precise statement of the result, first we introduce the following further notations.

For a $\mathrm{FO}[\tau_\Sigma]$-formula φ we will shortly write $\varphi(X_1, \ldots, X_k)$ to indicate that $\mathit{free}(\varphi) = \{X_1, \ldots, X_k\}$. It means φ has free set variables $X_1, \ldots, X_k$, but no free first-order variables. Such a formula *defines* a $+\omega$-picture language $L^{+\omega}(\varphi)$ over the extended alphabet $\Sigma \times \{0,1\}^k$ as follows: For a picture p over Σ and sets $A_1, \ldots, A_k \subseteq \mathrm{dom}(p)$, denoted by $\overline{A} = (A_1, \cdots, A_k)$, for short $\overline{A} = (A_i)_{1 \leqslant i \leqslant k}$, we write $(p, \overline{A})$ for the picture q over $\Sigma \times \{0,1\}^k$ such that $\mathrm{dom}(q) = \mathrm{dom}(p)$ and for each position $x = (i,j) \in \mathrm{dom}(p)$ the letter at position x in q is the tuple $(a, (\alpha_1, \ldots, \alpha_k))$ where $a \in \Sigma$ is the letter at position x in p, and $\alpha_\kappa \in \{0,1\}$ is defined for each $\kappa \in [k]$ via $\alpha_\kappa = 1 \iff x \in A_\kappa$. Using this notation, the $+\omega$-picture language *defined* by φ is

$$L^{+\omega}(\varphi) := \{(p, \overline{A}) \,:\, p \in \Sigma^{+\omega}, \overline{A} = (A_i)_{1 \leqslant i \leqslant k} \text{ with } A_i \subseteq \mathrm{dom}(p), (\underline{p}, \overline{A}) \models \varphi\}.$$

Accordingly, the language of *finite pictures*, which we considered in Chapter 1, can be defined by φ as follows:

$$L^{++}(\varphi) := \{(p, \overline{A}) \,:\, p \in \Sigma^{++}, \overline{A} = (A_i)_{1 \leqslant i \leqslant k} \text{ with } A_i \subseteq \mathrm{dom}(p), (\underline{p}, \overline{A}) \models \varphi\}.$$

Throughout the remainder of this section, a language L of (finite or) $+\omega$-pictures over the alphabet $\Sigma \times \{0,1\}^k$ is called *first-order definable* if there exists a formula $\varphi(X_1, \ldots, X_k) \in \mathrm{FO}[\tau_\Sigma]$ such that $L = L^{+\omega}(\varphi)$ (respectively, $L = L^{++}(\varphi)$).

The concept of *locally threshold testable picture languages* was already introduced in [64] and it can be adapted to the setting of $+\omega$-pictures. We let Γ be an extended alphabet of the form $\Sigma \times \{0,1\}^k$, for some $k \in \mathbb{N}$. Let $t \in \mathbb{N}_{\geqslant 1}$. For a (finite or $+\omega$)-picture p over Γ and a finite picture σ over $\hat{\Gamma}$, the *t-occurrence number* of σ in p, denoted by $\mathrm{occ}^t_\sigma(p)$, is defined by $\mathrm{occ}^t_\sigma(p) = i$, if $i < t$ and the number of occurrences of σ in $\hat{p}$ is exactly i, and $\mathrm{occ}^t_\sigma(p) = t$, if the number of occurrences of σ in $\hat{p}$ is $\geqslant t$.

Given a *threshold number* $t \in \mathbb{N}_{\geqslant 1}$ and a *subpicture size* $d \in \mathbb{N}_{\geqslant 1}$, two (finite or $+\omega$-)pictures p_1 and p_2 over Γ are called (d,t)-*equivalent*, denoted by $p_1 \sim_{d,t} p_2$, if and only if $\mathrm{occ}^t_\sigma(p_1) = \mathrm{occ}^t_\sigma(p_2)$ holds for all finite pictures σ over $\hat{\Gamma}$ of height and width $\leqslant d$. It means p_1 and p_2 are (d,t)-equivalent if and only if for each picture σ over $\hat{\Gamma}$ of height and width $\leqslant d$, there are either at least t occurrences of σ in both $\hat{p}_1$ and $\hat{p}_2$, or the numbers $(\leqslant t)$ of occurrences of σ in $\hat{p}_1$ and $\hat{p}_2$ coincide. It is straightforward to see that $\sim_{d,t}$ is an equivalence relation of finite index. A language L of (finite or $+\omega$-)pictures over Γ is called

- *locally d-testable with threshold t* if L is a union of $\sim_{d,t}$-equivalence classes;
- *locally threshold d-testable* if there exists a $t \in \mathbb{N}_{\geqslant 1}$ such that L is locally d-testable with threshold t;
- *locally threshold testable* if there exists a $d \in \mathbb{N}_{\geqslant 1}$ such that L is locally threshold d-testable.

Note that locally threshold d-testable languages are also locally threshold d'-testable, for every $d' \geqslant d$.

Theorem 3.3 of [64] states that a language of finite pictures is first-order definable if and only if it is locally threshold testable. The proof given in [64] can be applied to obtain the following.

Theorem 2.18. *Let Σ be an alphabet and let $k \in \mathbb{N}$. A language of $+\omega$-pictures over alphabet $\Sigma \times \{0,1\}^k$ is first-order definable if and only if it is locally threshold testable.*

Our overall goal throughout the remainder of this section is to prove Theorem 2.17, i.e., we are given an $\mathrm{EMSO}^\infty[\tau_\Sigma]$-sentence Φ, and we want to construct a Büchi-tiling system $\mathcal{S}_\Phi$ with $L^{+\omega}(\mathcal{S}_\Phi) = L^{+\omega}(\Phi)$. As Φ is an $\mathrm{EMSO}^\infty[\tau_\Sigma]$-sentence, Φ is of the form

$$\exists^\infty X_1 \cdots \exists^\infty X_{k'}\ \exists X_{k'+1} \cdots \exists X_k\ \varphi$$

where $0 \leqslant k' \leqslant k$ and $\varphi(X_1, \ldots, X_k)$ is an $\mathrm{FO}[\tau_\Sigma]$-formula.

According to Theorem 2.18, the formula $\varphi(X_1, \ldots, X_k)$ defines a locally threshold testable language of $+\omega$-pictures over the alphabet $\Sigma \times \{0,1\}^k$. Now the following lemma enables us to give the proof of Theorem 2.17.

Lemma 2.19. *Let Γ be an alphabet. Every locally threshold testable $+\omega$-picture language over Γ is Büchi-tiling recognizable.*

Remark 2.20. *Note that for translation of formulas in* $\mathrm{FO}[\tau_\Sigma]$ *to Büchi tiling systems we cannot use induction on the structure of* $\varphi \in \mathrm{FO}[\tau_\Sigma]$. *This is due to the fact that recognizable* $+\omega$*-picture languages are not closed under complement (See Proposition 2.11); however,* $\mathrm{FO}[\tau_\Sigma]$ *is closed under negation, i.e., whenever* $\varphi \in \mathrm{FO}[\tau_\Sigma]$, *then* $\neg\varphi \in \mathrm{FO}[\tau_\Sigma]$.

Before giving the proof of this lemma, let us first apply it to prove Theorem 2.17.

Proof of Theorem 2.17. Let Σ be an alphabet, and let $L \subseteq \Sigma^{+\omega}$ be a $+\omega$-picture language that is defined by an $\mathrm{EMSO}^\infty[\tau_\Sigma]$-sentence Φ. Let Φ be of the form

$$\exists^\infty X_1 \cdots \exists^\infty X_{k'}\ \exists X_{k'+1} \cdots \exists X_k\ \varphi$$

where $0 \leqslant k' \leqslant k$, and where $\varphi(X_1, \ldots, X_k)$ is a $\mathrm{FO}[\tau_\Sigma]$-formula. Let $L' \subseteq (\Sigma \times \{0,1\}^k)^{+\omega}$ be the $+\omega$-picture language defined by the first-order formula $\varphi(X_1, \ldots, X_k)$. By Theorem 2.18, L' is locally threshold testable; and by Lemma 2.19, L' is Büchi-tiling recognizable. Let $\mathcal{S}' = \big(\Sigma \times \{0,1\}^k, Q', \Theta', F'\big)$ be a Büchi-tiling system that recognizes L'. We construct a generalized Büchi-tiling system $\mathcal{S} = \big(\Sigma, Q, \Theta, \tilde{F}\big)$ that recognizes L. The construction is done in such a way that a run r of $\mathcal{S}$ on a $+\omega$-picture $p \in \Sigma^{+\omega}$ can be translated into a run r' of $\mathcal{S}'$ on $(p, \overline{A})$, where $\overline{A} = A_1, \ldots, A_k \subseteq \mathrm{dom}(p)$ is an assignment for the set variables $X_1, \ldots, X_k$. The run r is *accepted* if and only if r' is accepted; in addition, each of the sets $A_1, \ldots, A_{k'}$ is infinite. To ensure all this, we let $\mathcal{S}$ have the state space

$$Q \;=\; \{0,1\}^k \times Q' \times \{0,1\}^k.$$

The intended meaning is that if a position $x = (i,j)$ of a picture $p \in \Sigma^{+\omega}$ is assigned to a state of the form $(\overline{\alpha}, q', \overline{\gamma})$ with $\overline{\alpha} = (\alpha_1, \ldots, \alpha_k) \in \{0,1\}^k$ and $q' \in Q'$ and $\overline{\gamma} = (\gamma_1, \ldots, \gamma_k) \in \{0,1\}^k$, then we interpret this as follows:

1. $x \in A_\kappa$ for exactly those $\kappa \in [k]$ where $\alpha_\kappa = 1$, and
2. position x is assigned to the state $q' \in Q'$ of $\mathcal{S}'$, and
3. for each $\kappa \in [k]$, we have $\gamma_\kappa = 1$ if and only if x or some position in the same column as x, but below x, belongs to A_κ.

As acceptance condition of $\mathcal{S}'$ we can then choose $\tilde{F} = \{F_0, F_1, \ldots, F_{k'}\}$ with

$$F_0 \;=\; \{0,1\}^k \times F' \times \{0,1\}^k$$

and, for each $\kappa \in \{1, \ldots, k'\}$,

$$F_\kappa \;=\; \{0,1\}^k \times Q' \times H_\kappa$$

with $H_\kappa = \{\ (\gamma_1, \ldots, \gamma_k) \in \{0,1\}^k \ :\ \gamma_\kappa = 1\ \}$.

In the sequel, whenever we write $\overline{\beta} \in \{0,1\}^k$ for some symbol β, we mean $\beta_1, \ldots, \beta_k$ to refer to the components $1, \ldots, k$ of $\overline{\beta}$, in another word, $\overline{\beta} = (\beta_1, \ldots, \beta_k)$. For $\overline{\alpha}, \overline{\gamma} \in \{0,1\}^k$ we write $\overline{\gamma} \geqslant \overline{\alpha}$ if and only if $\gamma_\kappa \geqslant \alpha_\kappa$ for all $\kappa \in [k]$. The set Θ of tiles of $\mathcal{S}$ is obtained as follows:

• Consider all tiles in Θ' of the form

#	$\big((a, \overline{\alpha}),\ q\big)$
#	#

with $(a, \overline{\alpha}) \in \Sigma \times \{0,1\}^k$ and $q \in Q'$. Add to Θ the tile

#	$\big(a,\ (\overline{\alpha}, q, \overline{\alpha})\big)$
#	#

.

• Consider all tiles in Θ' of the form

#	$\big((a', \overline{\alpha}'),\ q'\big)$
#	$\big((a, \overline{\alpha}),\ q\big)$

with $(a, \overline{\alpha}) \in \Sigma \times \{0,1\}^k$, $(a', \overline{\alpha}') \in \Sigma \times \{0,1\}^k$ and $q, q' \in Q'$. Consider all $\overline{\gamma} \in \{0,1\}^k$ with $\overline{\gamma} \geqslant \overline{\alpha}$. Let $\overline{\gamma}'$ be the element of $\{0,1\}^k$ defined via $\gamma'_\kappa = \max\{\alpha'_\kappa, \gamma_\kappa\}$, for all $\kappa \in [k]$. Add to Θ the tile

#	$\big(a',\ (\overline{\alpha}', q', \overline{\gamma}')\big)$
#	$\big(a,\ (\overline{\alpha}, q, \overline{\gamma})\big)$

.

• Consider all tiles in Θ' of the form

#	#
#	$\big((a, \overline{\alpha}),\ q\big)$

with $(a, \overline{\alpha}) \in \Sigma \times \{0,1\}^k$ and $q \in Q'$. Consider all $\overline{\gamma} \in \{0,1\}^k$ with $\overline{\gamma} \geqslant \overline{\alpha}$, and add to Θ the tile

#	#
#	$\big(a,\ (\overline{\alpha}, q, \overline{\gamma})\big)$

.

• Consider all tiles in Θ' of the form

$\big((a, \overline{\alpha}),\ q\big)$	$\big((a', \overline{\alpha}'),\ q'\big)$
#	#

with $(a,\overline{\alpha}) \in \Sigma \times \{0,1\}^k$ and $(a',\overline{\alpha}') \in \Sigma \times \{0,1\}^k$ and $q, q' \in Q'$. Add to Θ the tile

$\big(a,\ (\overline{\alpha}, q, \overline{\alpha})\big)$	$\big(a',\ (\overline{\alpha}', q', \overline{\alpha}')\big)$
$\#$	$\#$

.

• Consider all tiles in Θ' of the form

$\big((a',\overline{\alpha}'),\ q'\big)$	$\big((b',\overline{\beta}'),\ p'\big)$
$\big((a,\overline{\alpha}),\ q\big)$	$\big((b,\overline{\beta}),\ p\big)$

with $(a,\overline{\alpha}),\ (a',\overline{\alpha}'),\ (b,\overline{\beta}),\ (b',\overline{\beta}') \in \Sigma \times \{0,1\}^k$ and $q, q', p, p' \in Q'$. Consider all $\overline{\gamma} \in \{0,1\}^k$ with $\overline{\gamma} \geqslant \overline{\alpha}$, and consider all $\overline{\delta} \in \{0,1\}^k$ with $\overline{\delta} \geqslant \overline{\beta}$. Let $\overline{\gamma}'$ be the element of $\{0,1\}^k$ defined via $\gamma'_\kappa = \max\{\alpha'_\kappa, \gamma_\kappa\}$, for all $\kappa \in [k]$. Let $\overline{\delta}'$ be the element of $\{0,1\}^k$ defined via $\delta'_\kappa = \max\{\beta'_\kappa, \delta_\kappa\}$, for all $\kappa \in [k]$. Add to Θ the tile

$\big(a',\ (\overline{\alpha}', q', \overline{\gamma}')\big)$	$\big(b',\ (\overline{\beta}', p', \overline{\delta}')\big)$
$\big(a,\ (\overline{\alpha}, q, \overline{\gamma})\big)$	$\big(b,\ (\overline{\beta}, p, \overline{\delta})\big)$

.

• Finally, consider all tiles in Θ' of the form

$\#$	$\#$
$\big((a,\overline{\alpha}),\ q\big)$	$\big((a',\overline{\alpha}'),\ q'\big)$

with $(a,\overline{\alpha}) \in \Sigma \times \{0,1\}^k$, $(a',\overline{\alpha}') \in \Sigma \times \{0,1\}^k$ and $q, q' \in Q'$. Consider all $\overline{\gamma} \in \{0,1\}^k$ with $\overline{\gamma} \geqslant \overline{\alpha}$, and consider all $\overline{\gamma}' \in \{0,1\}^k$ with $\overline{\gamma}' \geqslant \overline{\alpha}'$, and add to Θ the tile

$\#$	$\#$
$\big(a,\ (\overline{\alpha}, q, \overline{\gamma})\big)$	$\big(a',\ (\overline{\alpha}', q', \overline{\gamma}')\big)$

.

Finally, the construction of the generalized Büchi-tiling system $\mathcal{S}$ is complete. The construction was done in such a way that a run r of $\mathcal{S}$ on a picture $p \in \Sigma^{+\omega}$ can be translated into a mapping $r' : \mathrm{dom}(p) \to Q'$ and an assignment $\overline{A} = A_1, \ldots, A_k \subseteq \mathrm{dom}(p)$ for the set variables $X_1, \ldots, X_k$, such that r' is a run of $\mathcal{S}'$ on the picture $p' = (p, \overline{A})$, and vice versa. It is straightforward to verify that $p \in \Sigma^{+\omega}$ is accepted by $\mathcal{S}$ if, and only if, there exists an assignment $\overline{A} = A_1, \ldots, A_k \subseteq \mathrm{dom}(p)$ such $(p, \overline{A})$ is accepted by $\mathcal{S}'$ and each of the sets $A_1, \ldots, A_{k'}$ is infinite. Recall that $\mathcal{S}'$ recognizes the language

$$L' \ = \ \{\ (p,\overline{A}) \ :\ p \in \Sigma^{+\omega},\ \overline{A} = A_1, \ldots, A_k \subseteq \mathrm{dom}(p),\ (\underline{p},\overline{A}) \models \varphi\ \}.$$

Thus, $p \in \Sigma^{+\omega}$ is accepted by $\mathcal{S}$ if, and only if, $\underline{p} \models \Phi$. From Proposition 2.8 we know that generalized Büchi-tiling systems can be transformed into

equivalent Büchi-tiling systems. Hence, the language $L^{+\omega}(\Phi)$ is Büchi-tiling recognizable. This completes the proof of Theorem 2.17.

□

All that remains to be done is to give the proof of Lemma 2.19. Recall that Lemma 2.19 states that every locally threshold testable $+\omega$-picture language L over an alphabet Γ is Büchi-tiling recognizable. The first steps in the proof of this lemma are identical to the steps taken in [64] for proving the according result concerning languages of finite pictures: The first obstacle to overcome is that the notion of locally threshold testability for threshold t and size d takes into account the number of occurrences of pictures of arbitrary height and width $\leqslant d$. As a remedy, one decomposes locally threshold testable languages into finite unions of projections of languages that are *locally strictly threshold testable* in the following sense: Given a threshold number $t \in \mathbb{N}_{\geqslant 1}$ and a square size $d \in \mathbb{N}_{\geqslant 1}$, two (finite or $+\omega$-)pictures p_1 and p_2 over Γ are called *strictly (d,t)-equivalent*, denoted by $p_1 \simeq_{d,t} p_2$, if $\mathrm{occ}^t_\sigma(p_1) = \mathrm{occ}^t_\sigma(p_2)$ holds for all pictures $\sigma \in \hat{\Gamma}^{d,d}$. In other words, p_1 and p_2 are strictly (d,t)-equivalent if, and only if, for each square picture σ of size (d,d) over $\hat{\Gamma}$, there are either at least t occurrences of σ in both $\hat{p}_1$ and $\hat{p}_2$, or the numbers ($\leqslant t$) of occurrences of σ in $\hat{p}_1$ and $\hat{p}_2$ coincide. It is straightforward to see that $\simeq_{d,t}$ is an equivalence relation of finite index.

A language L of (finite or $+\omega$-)pictures over Γ is called

- *locally strictly d-testable with threshold t* if L is a union of $\simeq_{d,t}$-equivalence classes
- *locally strictly threshold d-testable* if there exists a $t \in \mathbb{N}_{\geqslant 1}$ such that L is locally strictly d-testable with threshold t.
- *locally strictly threshold testable* if there exists a $d \in \mathbb{N}_{\geqslant 1}$ such that L is locally strictly d-testable.

Recall that $p_1 \simeq_{d,t} p_2$ holds for two pictures p_1 and p_2 over Γ if and only if $\mathrm{occ}^t_\sigma(p_1) = \mathrm{occ}^t_\sigma(p_2)$ is true for all square pictures σ of size (d,d), i.e., $\sigma \in \hat{\Gamma}^{d,d}$.

Let $d \in \mathbb{N}$ with $d \geqslant 3$. Lemma 3.7 of [64] states that every locally threshold d-testable language L of *finite* pictures can be decomposed into $L_1 \cup \cdots \cup L_{d-2}$ where for each $i \leqslant d-2$, the language L_i is a *locally strictly $(i+2)$-testable* language of finite pictures of height and width $\geqslant i$. The proof given in [64] can be used to obtain the following.

Lemma 2.21. *Let Γ be an alphabet and let $d \in \mathbb{N}$ with $d \geqslant 3$. Every locally threshold d-testable language $L \subseteq \Gamma^{+\omega}$ can be decomposed into $L_1 \cup \cdots \cup L_{d-2}$, where for each $i \leqslant d-2$, the language L_i is a projection of a locally strictly threshold $(i+2)$-testable $+\omega$-picture languages and contains only $+\omega$-pictures of height $\geqslant i$.*

By applying this lemma and also the fact that the class of Büchi-tiling recognizable $+\omega$-picture languages is closed under union and projection (Proposition 2.9), we can see that for proving Lemma 2.19 it suffices to show the following Lemma. Below, for an integer h and an alphabet Σ, we write $\Sigma^{+\omega}_{\geqslant h}$ to denote the set of all $+\omega$-pictures over Σ of height $\geqslant h$.

Lemma 2.22. *Let Σ be an alphabet and let $d \in \mathbb{N}$ with $d \geqslant 3$. Every locally strictly threshold d-testable language $L \subseteq \Sigma^{+\omega}_{\geqslant d-2}$ is Büchi-tiling recognizable.*

The remainder of this section is devoted to the proof of Lemma 2.22. Note that once the proof of this lemma is finished, the proof of Theorem 2.17, and as a result Theorem 2.14, is complete. An obstacle for proving Lemma 2.22 is that the notion of locally strictly threshold d-testability takes into account occurrences of subpictures of size d (i.e., subpictures $\sigma \in \hat{\Sigma}^{d,d}$), while Büchi-tiling systems are defined by using tiles of size 2 only. To overcome this obstacle, one considers an extended version of Büchi-tiling systems that use tiles of size d, and shows that these can be translated into equivalent Büchi-tiling systems that use tiles of size only 2.

Definition 2.23. *Let Σ be an alphabet and let $d \in \mathbb{N}$ with $d \geqslant 2$. A d-tile is a picture of size (d,d) over $\hat{\Gamma}$, for some alphabet Γ. A Büchi-d-tiling system is a 4-tuple $\mathcal{S} = (\Sigma, Q, \Theta, F)$ where Σ and Q are alphabets, $F \subseteq Q$, and $\Theta \subseteq \hat{\Gamma}^{d,d}$ for $\Gamma = \Sigma \times Q$.*

A *run* of $\mathcal{S}$ on a picture $p \in \Sigma^{+\omega}_{\geqslant d-2}$ is a picture r over Q of the same size as p, such that the picture $q = (p \times r)$ has the following property: every subpicture of size (d,d) of $\hat{q}$ belongs to Θ, i.e., $T_{d,d}(\hat{q}) \subseteq \Theta$. A run r of $\mathcal{S}$ on p is *accepting*, if $\inf_1(r) \cap F \neq \emptyset$. A picture $p \in \Sigma^{+\omega}_{\geqslant d-2}$ is *accepted* by $\mathcal{S}$ if there exists an accepting run of $\mathcal{S}$ on p. The $+\omega$-picture language *recognized* by $\mathcal{S}$ is the set $L^{+\omega}(\mathcal{S})$ of all pictures $p \in \Sigma^{+\omega}_{\geqslant d-2}$ that are accepted by $\mathcal{S}$.

Note that Büchi-2-tiling systems are exactly the Büchi-tiling systems introduced in Definition 2.7. We will prove the following two lemmas:

Lemma 2.24. *Let Σ be an alphabet and let $d \in \mathbb{N}$ with $d \geqslant 3$. For every locally strictly threshold d-testable language $L \subseteq \Sigma^{+\omega}_{\geqslant d-2}$ there exists a Büchi-d-tiling system $\mathcal{S}$ such that $L = L^{+\omega}(\mathcal{S})$.*

Lemma 2.25. *Let Σ be an alphabet, let $d \in \mathbb{N}$ with $d \geqslant 3$, and let $\mathcal{S}$ be a Büchi-d-tiling system. There exists a Büchi-$(d-1)$-tiling system $\mathcal{S}'$ such that $L^{+\omega}(\mathcal{S}') = L^{+\omega}(\mathcal{S})$.*

Note that Lemma 2.22 is an immediate consequence of Lemma 2.24 and Lemma 2.25. The proof of Lemma 2.25 proceeds along a well-known construction that has already been used for generalized versions of tiling systems for finite picture languages (see, e.g., the main result of [75], or see the proof of Lemma 3.10 in [64] for a transformation from d-tiles to 2-tiles). For the sake of completeness, we include a proof at the end of this section. The proof of Lemma 2.24 is a bit more involved. It starts off in the same way as the proof of Lemma 3.9 in [64], but has to employ a different technique for counting the number of occurrences of a subpicture of size (d, d) in a given picture. Below, first we give the proof of Lemma 2.24.

Proof of Lemma 2.24. Let $d \in \mathbb{N}$ with $d \geqslant 3$, let $t \in \mathbb{N}_{\geqslant 1}$, and let $L \subseteq \Sigma^{+\omega}_{\geqslant d-2}$ be locally strictly d-testable with threshold t. I.e., L is a union of $\simeq_{d,t}$-equivalence classes. Let $D = \hat{\Sigma}^{d,d}$. Note that every $\simeq_{d,t}$-equivalence class can be represented by a function $\gamma : D \to \{0, \ldots, t\}$: the equivalence class represented by γ consists of all pictures p where $\mathrm{occ}^t_\sigma(p) = \gamma(\sigma)$ for every $\sigma \in D$. For each (finite or $+\omega$-)picture p over $\hat{\Sigma}$ we let γ_p be the function from D to $\{0, \ldots, t\}$, and for every $\sigma \in D$, it is defined by $\gamma_p(\sigma) = i$, if $i < t$ and the number of occurrences of σ in p is exactly i, and $\gamma_p(\sigma) = t$, if the number of occurrences of σ in p is $\geqslant t$. It means, γ_p contains information on the number of occurrences (up to threshold t) of every picture $\sigma \in D$. For the remainder of this proof, the function γ_p will be called the *type* of p. Note the subtle difference between the definition of $\gamma_p(\sigma)$ and $\mathrm{occ}^t_\sigma(p)$: While $\gamma_p(\sigma)$ counts the number of occurrences of σ in p, $\mathrm{occ}^t_\sigma(p)$ counts the number of occurrences of σ in $\hat{p}$, i.e., in the picture obtained by surrounding p with #-symbols from the left, the top, and the bottom. Thus, $\mathrm{occ}^t_\sigma(p) = \gamma_{\hat{p}}(\sigma)$. Furthermore, note that for every $+\omega$-picture $p \in \Sigma^{+\omega}$ there exists a $j_0 \in \mathbb{N}_{\geqslant 1}$ such that for all columns $j \geqslant j_0$ the $+\omega$-picture $\hat{p}$ has the same type as the finite picture $\hat{p}[1, j]$, i.e., $\gamma_{\hat{p}} = \gamma_{\hat{p}[1,j]}$ for all $j \geqslant j_0$.

Let $\mathfrak{C}$ be the set of all functions $\gamma : D \to \{0, \ldots, t\}$. By assumption, L is a union of $\simeq_{d,t}$-equivalence classes. In other words, there is a set $C \subseteq \mathfrak{C}$ such that for every picture $p \in \Sigma^{+\omega}$ we have $p \in L \iff \gamma_{\hat{p}} \in C$. Our goal is to construct a Büchi-d-tiling system $\mathcal{S}$ which accepts an input picture $p \in \Sigma^{+\omega}_{\geqslant d-2}$ if, and only if, $\gamma_{\hat{p}[1,j]} \in C$ is true for infinitely many columns j; note that the latter is true if, and only if, $\gamma_{\hat{p}} \in C$. We let $\mathcal{S} = (\Sigma, Q, \Theta, F)$,

where $Q = \mathfrak{C} \times \mathfrak{C}$ and $F = \mathfrak{C} \times C$. We will construct Θ in such a way that for every picture $p \in \Sigma^{+\omega}_{\geqslant d-2}$ there exists exactly one run r, and this run has the following property: For any column $j \geqslant d-1$, the entry r_{1j} of r in row 1 and column j is of the form (γ, γ'), where

- $\gamma' = \gamma_{q'}$ for the picture $q' = \hat{p}[1, j+1]$. Note that q' is the picture obtained by surrounding $p[1, j]$ with #-symbols from the left, the top, and the bottom.
- $\gamma = \gamma_q$ for the picture q which consists of the last d columns of q'.

In general, for any column $j \geqslant d-1$ and any row $i \in \{1, \ldots, m-d+3\}$, the entry r_{ij} of r in row i and column j shall be of the form (γ, γ'), where

- $\gamma' = \gamma_{q'}$ for the picture $q' = \hat{p}^i_{m+2}[1, j+1]$.
- $\gamma = \gamma_q$ for the picture q which consists of the last d columns of q'.

And for all i, j with $j < d-1$ or $i > m-d+3$, the entry r_{ij} of r in row i and column j shall be (γ_0, γ_0), where $\gamma_0(\sigma) = 0$ for all $\sigma \in D$. By our choice of $F = \mathfrak{C} \times C$, such a run will be accepting if, and only if, for infinitely many columns j the picture $q' = \hat{p}[1, j+1]$ has a type $\gamma_{q'}$ that belongs to C. Since $\gamma_{\hat{p}} = \gamma_{\hat{p}[1,j]}$ for all sufficiently large j, this means that a run will be accepting if, and only if, $\gamma_{\hat{p}} \in C$ — and this is the case if, and only if, $p \in L$. Thus, all that is needed to finish the proof of Lemma 2.24 is to construct the set Θ in such a way that runs have the property described above.

The idea underlying the construction of Θ is similar as in [64]: We perform a scan of the given $+\omega$-picture $\hat{p}$ by using a square window of size (d, d) and count (up to the threshold t) how many times we see, through the window, each picture $\sigma \in D$. The scanning is carried out along all vertical stripes of width d. For each vertical stripe, we proceed from bottom to top and record the results of the threshold counting in the first component of the state at position $x = (2, d)$ of a tile. The second component of the state at position x combines the first component of x's state with the second component of x's left neighbour's state. For giving the precise definition of the set Θ, the following notation will be useful: Let $\Gamma = \Sigma \times Q$. For a d-tile $T \in \hat{\Gamma}^{d,d}$ we write $\sigma(T)$ (resp., $\rho(T)$) for the picture of size (d, d) over $\hat{\Sigma}$ (resp., $\hat{Q}$) obtained from T by replacing every entry of the form $(a, q) \in \Sigma \times Q$ with a (resp., q). The set Θ of d-tiles of $\mathcal{S}$ is defined as follows:

• Add to Θ all tiles $T \in \hat{\Gamma}^{d,d}$ that satisfy the following conditions:

1. The first column and the bottom row of T only contain the border symbol # (i.e., $T_{i1} = T_{dj} = \#$, for all $i, j \in [d]$).

2. The entry in row 2 and column d of $\rho(T)$ is (γ, γ), where γ is the particular element in $\mathfrak{C}$ with $\gamma(\sigma(T)) = 1$ and $\gamma(\sigma') = 0$ for all $\sigma' \in D$ with $\sigma' \neq \sigma(T)$.

3. The entry in row 2 and each column $j \in \{2, \ldots, d-1\}$ of $\rho(T)$ is (γ_0, γ_0).

4. For every $i \in \{3, \ldots, d-1\}$ and every $j \in \{2, \ldots, d\}$, the entry in row i and column j of $\rho(T)$ is (γ_0, γ_0).

5. Either, the entry of $\rho(T)$ in row 1 is # in all columns $j \in \{2, \ldots, d-1\}$, or it is (γ_0, γ_0) in all columns $j \in \{2, \ldots, d-1\}$ (i.e., either $\rho(T)_{1j} = \#$ for all $j \in \{2, \ldots, d-1\}$, or $\rho(T)_{1j} = (\gamma_0, \gamma_0)$ for all $j \in \{2, \ldots, d-1\}$).

Note that we do not impose any restriction on the entry T_{1d} in row 1 and column d here.

• Add to Θ all tiles $T \in \hat{\Gamma}^{d,d}$ that satisfy the following conditions:

1. The first column of T only contains the border symbol #, and the bottom row of T contains no border symbol in columns > 1 (i.e., $T_{i1} = \#$ for all $i \in [d]$, and $T_{dj} \neq \#$, for all $j \in \{2, \ldots, d\}$).

2. There are $\gamma, \delta \in \mathfrak{C}$ such that the entry in row 2 and column d of $\rho(T)$ is of the form (γ, γ), the entry in row 3 and column d of $\rho(T)$ is of the form (δ, δ), and the following is true: $\gamma(\sigma(T)) = \min\{t,\ \delta(\sigma(T))+1\}$, and $\gamma(\sigma') = \delta(\sigma')$ for all $\sigma' \in D$ with $\sigma' \neq \sigma(T)$.

3. For all $i \in \{2, \ldots, d\}$ and all $j \in \{2, \ldots, d-1\}$, the entry in row i and column j of $\rho(T)$ is (γ_0, γ_0).

4. Either, the entry of $\rho(T)$ in row 1 is # in all columns $j \in \{2, \ldots, d-1\}$, or it is (γ_0, γ_0) in all columns $j \in \{2, \ldots, d-1\}$.

• Add to Θ all tiles $T \in \hat{\Gamma}^{d,d}$ that satisfy the following conditions:

1. The bottom row of T contains only border symbols #, and the first column of T contains no border symbol in rows $2, \ldots, d-1$ (i.e., $T_{dj} = \#$ for all $j \in [d]$, and $T_{i1} \neq \#$ for all $i \in \{2, \ldots, d-1\}$).

2. There are $\gamma, \gamma', \beta, \beta' \in \mathfrak{C}$ such that the entry in row 2 and column $d-1$ of $\rho(T)$ is (β, β'), the entry in row 2 and column d of $\rho(T)$ is (γ, γ'), and the following two conditions are satisfied:

- ⋆ γ is the particular element in $\mathfrak{C}$ with $\gamma(\sigma(T)) = 1$ and $\gamma(\sigma') = 0$, for every $\sigma' \in D$ with $\sigma' \neq \sigma$.
- ⋆ $\gamma' = \beta' +_t \gamma$. I.e., $\gamma'(\sigma) = \min\{t,\ \beta'(\sigma) + \gamma(\sigma)\}$, for all $\sigma \in D$.

3. For all $i \in \{3, \ldots, d{-}1\}$ and all $j \in [d]$, the entry in row i and column j of $\rho(T)$ is (γ_0, γ_0).

• Add to Θ all tiles $T \in \hat{\Gamma}^{d,d}$ that satisfy the following conditions:

1. The bottom row of T contains no border symbol #, and the first column of T contains no border symbol in rows $2, \ldots, d$ (i.e., $T_{dj} \neq \#$ for all $j \in [d]$, and $T_{i1} \neq \#$ for all $i \in \{2, \ldots, d\}$).

2. There are $\gamma, \gamma', \beta, \beta', \delta, \delta' \in \mathfrak{C}$ such that the entry in row 2 and column $d{-}1$ of $\rho(T)$ is (β, β'), the entry in row 2 and column d of $\rho(T)$ is (γ, γ'), the entry in row 3 and column d of $\rho(T)$ is (δ, δ'), and the following two conditions are satisfied:

   - ⋆ $\gamma(\sigma(T)) = \min\{t,\ \delta(\sigma(T)) + 1\}$, and $\gamma(\sigma') = \delta(\sigma')$ for all $\sigma' \in D$ with $\sigma' \neq \sigma$.
   - ⋆ $\gamma' = \beta' +_t \gamma$. I.e., $\gamma'(\sigma) = \min\{t,\ \beta'(\sigma){+}\gamma(\sigma)\}$, for all $\sigma \in D$.

Finally, the construction of the Büchi-d-tiling system $\mathcal{S}$ is complete. The construction was done in such a way that for every picture $p \in \Sigma^{+\omega}$ there is exactly one run of $\mathcal{S}$ on p, and this run has the property described at the beginning of the proof (to verify this, one can proceed by a nested induction that considers all columns $j = d{-}1, d, d{+}1, d{+}2, \ldots$ and for each j, one considers rows $i = m{-}d{+}3, m{-}d{+}2, \ldots, 1$, where m is the height of p). This completes the proof of Lemma 2.24.

□

As mentioned already, the proof of Lemma 2.25 follows a construction used already, e.g., in [75, 64]. In the following, we have adapted this construction to the context of Büchi-d-tiling systems.

Proof of Lemma 2.25. Let Σ be an alphabet, let $d \in \mathbb{N}$ with $d \geqslant 3$, and let $\mathcal{S} = (\Sigma, Q, \Theta, F)$ be a Büchi-d-tiling system. Let $L = L^{+\omega}(\mathcal{S})$ be the $+\omega$-picture language recognized by $\mathcal{S}$ (in particular, $L \subseteq \Sigma^{+\omega}_{\geqslant d-2}$). The goal is to construct a Büchi-$(d{-}1)$-tiling system $\mathcal{S}'$ with $L^{+\omega}(\mathcal{S}') = L^{+\omega}(\mathcal{S})$. We proceed in a similar way as Latteux and Simplot in [75]. Let $\Gamma = \Sigma \times Q$. Recall that $\Theta \subseteq \hat{\Gamma}^{d,d}$. As in [75], we consider the *extended alphabet*

$$ext(\Gamma) \ = \ \hat{\Gamma}^{3,3},$$

and we define a mapping θ from $\Gamma^{+\omega}$ to $ext(\Gamma)^{+\omega}$ which maps every $+\omega$-picture q over Γ to the $+\omega$-picture $\theta(q)$ over $ext(\Gamma)$ that has the same height $m \in \mathbb{N}_{\geqslant 1}$ as q, and where the "extended letter" of $\theta(q)$ at any position $x \in \mathrm{dom}(q)$ is the subpicture of $\hat{q}$ of size $(3,3)$ whose *middle position* corresponds to x. Note that position (i,j) of q corresponds to position $(i{+}1, j{+}1)$ of $\hat{q}$. Thus, for all positions $x = (i,j)$ with $i \in \{1,\ldots,m\}$ and $j \in \mathbb{N}_{\geqslant 1}$ we have

$$\theta(q)_{ij} \quad = \quad \begin{array}{|c|c|c|} \hline \hat{q}_{ij} & \hat{q}_{i(j+1)} & \hat{q}_{i(j+2)} \\ \hline \hat{q}_{(i+1)j} & \hat{q}_{(i+1)(j+1)} & \hat{q}_{(i+1)(j+2)} \\ \hline \hat{q}_{(i+2)j} & \hat{q}_{(i+2)(j+1)} & \hat{q}_{(i+2)(j+2)} \\ \hline \end{array} .$$

We also define for all $\mu, \nu \in \{1,2,3\}$ a mapping $\pi_{\mu\nu} : \widehat{ext(\Gamma)} \to \hat{\Gamma}$ by $\pi_{\mu\nu}(c) = c_{\mu\nu}$ for every $c \in ext(\Gamma)$, and $\pi_{\mu\nu}(\#) = \#$. We let $\pi = \pi_{22}$. As usual, we lift π to a mapping from pictures over the alphabet $\widehat{ext(\Gamma)}$ to pictures over the alphabet $\hat{\Gamma}$ by applying π to every position of a given picture over $\widehat{ext(\Gamma)}$. Clearly, we have

$$\pi\big(\theta(q)\big) \;=\; q, \qquad \text{for every picture } q \text{ over } \Gamma.$$

Now let $K = \theta(\Gamma^{+\omega}_{\geqslant d-2})$. I.e., K is the set of all $+\omega$-pictures q' of height $\geqslant (d{-}2)$ over the extended alphabet $ext(\Gamma)$, for which there exists a $+\omega$-picture q over Γ, such that $q' = \theta(q)$.

Claim 2.26. *There is a set Δ_K of $(d{-}1)$-tiles over $\widehat{ext(\Gamma)}$ such that*

$$K \quad = \quad \{\; q' \in ext(\Gamma)^{+\omega} \;:\; T_{d-1,d-1}(\widehat{q'}) \subseteq \Delta_K \;\} \; .$$

Proof of Claim 2.26. We choose Δ_K to contain all subpictures of size $(d{-}1, d{-}1)$ of $\widehat{q'}$, for all pictures $q' \in K$. I.e.,

$$\Delta_K \quad = \quad \bigcup_{q' \in K} T_{d-1,d-1}(\widehat{q'}) \, .$$

We let $L^{+\omega}(\Delta_K) = \{\; q' \in ext(\Gamma)^{+\omega} \;:\; T_{d-1,d-1}(\widehat{q'}) \subseteq \Delta_K \;\}$.Obviously, our choice of Δ_K implies that $K \subseteq L^{+\omega}(\Delta_K)$. For the opposite direction, i.e., the statement $L^{+\omega}(\Delta_K) \subseteq K$ let q' be an arbitrary element of $L^{+\omega}(\Delta_K)$ and let $q = \pi(q')$. Proceeding by induction on the columns $j = 1, 2, 3, \ldots$ and, for each fixed column j, by induction on the rows $i = 1, 2, \ldots, \ell_1(\widehat{q'})$, it is straightforward to verify that $q \in \Gamma^{+\omega}_{\geqslant d-2}$ and $q' = \theta(q)$. This completes the proof of Claim 2.26. $\square$

Now, let Δ_K be the set of $(d{-}1)$-tiles over $\widehat{ext(\Gamma)}$ provided by Claim 2.26. With every $(d{-}1)$-tile t over $\widehat{ext(\Gamma)}$ we associate a picture $\gamma(t)$ of size $(d{+}1, d{+}1)$ over $\hat{\Gamma}$ that is obtained by surrounding $\pi(t)$ with information on the left, right, upper and lower border that is included in the extended letters of t in positions on the left, right, upper, and lower border of t. Precisely, we define $\gamma(t)$ as follows:

- For all $i, j \in \{1, \ldots, d{-}1\}$, the entry in row $i{+}1$ and column $j{+}1$ of $\gamma(t)$ is
$$\gamma(t)_{i+1,j+1} \quad = \quad \pi_{22}(t_{ij}) \ .$$
- For all $j \in \{1, \ldots, d{-}1\}$, the entry in row 1 and column $j{+}1$ of $\gamma(t)$ is
$$\gamma(t)_{1,j+1} \quad = \quad \pi_{12}(t_{1j}) \ .$$
- For all $j \in \{1, \ldots, d{-}1\}$, the entry in row $d{+}1$ and column $j{+}1$ of $\gamma(t)$ is
$$\gamma(t)_{d+1,j+1} \quad = \quad \pi_{32}(t_{(d-1)j}) \ .$$
- For all $i \in \{1, \ldots, d{-}1\}$, the entry in row $i{+}1$ and column 1 of $\gamma(t)$ is
$$\gamma(t)_{i+1,1} \quad = \quad \pi_{21}(t_{i1}) \ .$$
- For all $i \in \{1, \ldots, d{-}1\}$, the entry in row $i{+}1$ and column $d{+}1$ of $\gamma(t)$ is
$$\gamma(t)_{i+1,d+1} \quad = \quad \pi_{23}(t_{i(d-1)}) \ .$$
- The entries in the four corners of $\gamma(t)$ are
$$\begin{array}{rclcrcl} \gamma(t)_{11} & = & \pi_{11}(t_{11}) & , & \gamma(t)_{1(d+1)} & = & \pi_{13}(t_{1(d-1)}) \\ \gamma(t)_{(d+1)1} & = & \pi_{31}(t_{(d-1)1}) & , & \gamma(t)_{(d+1)(d+1)} & = & \pi_{33}(t_{(d-1)(d-1)}) \end{array}$$

Recall that, by assumption, we are given a set Θ of d-tiles over $\hat{\Gamma}$. We let Δ'_K be the set of all tiles $t \in \Delta_K$, where all subpictures of $\gamma(t)$ of size (d, d) are included in Θ. I.e.,
$$\Delta'_K \quad = \quad \{ \ t \in \Delta_K \ : \ T_{d,d}(\gamma(t)) \subseteq \Theta \ \} \, .$$
In particular, Δ'_K is a set of $(d{-}1)$-tiles over $\widehat{ext(\Gamma)}$.

Claim 2.27. *For the languages*
$$L_1 \quad = \quad L^{+\omega}(\Delta'_K) \quad = \quad \{ \ q' \in ext(\Gamma)^{+\omega} \ : \ T_{d-1,d-1}(\widehat{q'}) \subseteq \Delta'_K \ \}$$
and
$$L_2 \quad = \quad L^{+\omega}(\Theta) \quad = \quad \{ \ q \in \Gamma^{+\omega} \ : \ T_{d,d}(\hat{q}) \subseteq \Theta \ \}$$
we have $L_1 = \theta(L_2)$.

Proof of Claim 2.27. "$\subseteq$": Let $q' \in L_1$. By Claim 2.26 we know that there exists a $q \in \Gamma^{+\omega}_{\geqslant d-2}$ such that $q' = \theta(q)$. Our choice of Δ'_K implies that $T_{d,d}(\hat{q}) \subseteq \Theta$. Thus, $q \in L_2$, and hence $q' \in \theta(L_2)$.

"$\supseteq$": Let $q \in L_2$, and let $q' = \theta(q)$. Since $q \in L_2$, we know that $T_{d,d}(\hat{q}) \subseteq \Theta$. Our choice of Δ'_K implies that $T_{d-1,d-1}(\widehat{q'}) \subseteq \Delta'_K$. Thus, $q' \in L_1$.
This completes the proof of Claim 2.27. □

Now, we are ready to finish the proof of Lemma 2.25. Recall that, by assumption, we are given a Büchi-d-tiling system $\mathcal{S} = (\Sigma, Q, \Theta, F)$, and our goal is to construct a Büchi-$(d{-}1)$-tiling system $\mathcal{S}' = (\Sigma, Q', \Theta', F')$ that recognizes the same $+\omega$-picture language as $\mathcal{S}$. We choose $Q' = \mathit{ext}(\Gamma)$, we let $F' = \{c \in \mathit{ext}(\Gamma) : \pi(c) \in \Sigma \times F\}$, and we let

$$\Theta' \quad = \quad \{\, \tilde{t} \;:\; t \in \Delta'_K \,\},$$

where for all $i, j \in [d{-}1]$ the entry in row i and column j of $\tilde{t}$ is defined by $\tilde{t}_{ij} = \big(a, t_{ij}\big)$, if if $a \in \Sigma$ and there is a $q \in Q$ such that $\pi(t_{ij}) = (a, q)$, and $\tilde{t}_{ij} = \#$, otherwise. Note that for all $+\omega$-pictures p over Σ, every run r of $\mathcal{S}$ on p can be translated into a run r' of $\mathcal{S}'$ on p via $r' = \theta(p \times r)$; and r is accepting iff r' is accepting. Vice versa, every run r' of $\mathcal{S}'$ on p can be translated into a run r of $\mathcal{S}$ on p: Just consider the picture $\pi(r')$ and replace every entry of the form $(a, q) \in \Sigma \times Q$ with the entry q. Again, r is accepting iff r' is accepting. In summary, we obtain that $\mathcal{S}'$ accepts exactly the same pictures $p \in \Sigma^{+\omega}$ as $\mathcal{S}$. This completes the proof of Lemma 2.25. □

To conclude, note that we have proved Lemmas 2.24 and 2.25. Together with Proposition 2.8, this leads to a proof of Lemma 2.22. Together with Lemma 2.21 and Proposition 2.9, this yields a proof of Lemma 2.19. This finishes the proof of Theorem 2.17. Finally, the proof of Theorem 2.14, is an immediate result of Theorems 2.16 and 2.17.

Now we want to give the complete proof of Proposition 2.13, in which we claimed EMSO$[\tau_\Sigma]$ is not the appropriate tool to be applied for logical characterization of $+\omega$-picture languages.

Proof of Proposition 2.13. Obviously, L in the first statement is defined by the FO$[\tau_\Sigma]$-sentence $\exists x\, P_a(x) \wedge \neg\exists y(y S_h x \wedge x S_v y)$. The proof that L is not tiling recognizable can be taken verbatim from the proof of Proposition 2.5.

The Büchi-tiling recognizability of L', in the second statement, was already observed in Proposition 2.5. For proving that L' is not EMSO[τ_Σ]-definable, one can use a standard tool from mathematical logic: a Hanf-locality argument (cf., for example, Theorem 2.4.1 in [49] (Hanf's Theorem) and the application of Hanf's Theorem given in Proposition 2.4.5 in [49]). Since we already have available Theorem 2.18, it is not necessary to make an explicit use of Hanf's Theorem. We will use Theorem 2.18 instead. For contradiction, let us assume that L' is definable by an EMSO[τ_Σ]-sentence Φ. Let Φ be of the form

$$\exists X_1 \cdots \exists X_k \ \varphi$$

for a FO[τ_Σ]-formula $\varphi(X_1, \ldots, X_k)$. Let L'' be the $+\omega$-picture language over $\Gamma = \Sigma \times \{0,1\}^k$ that is defined by $\varphi(X_1, \ldots, X_k)$. According to Theorem 2.18, L'' is locally threshold testable. I.e., there are numbers $d, t \in \mathbb{N}_{\geqslant 1}$ such that L'' is a union of $\sim_{d,t}$-equivalence classes. We use similar notation as in the proof of Lemma 2.24, but now consider $\sim_{d,t}$-equivalence classes (instead of the $\simeq_{d,t}$-equivalence classes considered in the proof of Lemma 2.24). Let D be the set of all finite pictures over $\hat{\Gamma}$ of height and width $\leqslant d$. Note that every $\sim_{d,t}$-equivalence class can be represented by a function $\gamma : D \to \{0, \ldots, t\}$: the equivalence class represented by γ consists of all pictures p where $\mathrm{occ}^t_\sigma(p) = \gamma(\sigma)$ for every $\sigma \in D$. For each (finite or $+\omega$-)picture p over $\hat{\Gamma}$ we let γ_p be the function from D to $\{0, \ldots, t\}$, which, for every $\sigma \in D$, is defined by $\gamma_p(\sigma) = i$, if $i < t$ and the number of occurrences of σ in p is exactly i, and $\gamma_p(\sigma) = t$, if the number of occurrences of σ in p is $\geqslant t$.

For the remainder of this proof, the function γ_p will be called the *type* of p. Note the subtle difference between the definition of $\gamma_p(\sigma)$ and $\mathrm{occ}^t_\sigma(p)$: while $\gamma_p(\sigma)$ counts the number of occurrences of σ in p, $\mathrm{occ}^t_\sigma(p)$ counts the number of occurrences of σ in $\hat{p}$, i.e., in the picture obtained by surrounding p with #-symbols. Thus, $\mathrm{occ}^t_\sigma(p) = \gamma_{\hat{p}}(\sigma)$. Note that for every $+\omega$-picture $q \in \Gamma^{+\omega}$ there exists a $j_0 \in \mathbb{N}_{\geqslant 1}$ such that for all columns $j \geqslant j_0$ the $+\omega$-picture $\hat{q}$ has the same type as the finite picture $\hat{q}[1, j]$, i.e., $\gamma_{\hat{q}} = \gamma_{\hat{q}[1,j]}$. Now, to conclude the proof of the proposition, choose $n > (2^{kd}{+}1) \cdot (d{+}1)$. Let p be the $+\omega$-picture of height 1 that corresponds to the ω-word $(b^n a)^\omega$. Clearly, p contains an infinite number of a's and therefore belongs to the language L'. By assumption, $L' = L^{+\omega}(\Phi)$. Hence, $\underline{p} \models \Phi$. It means, there are sets $\overline{A} = A_1, \ldots, A_k \subseteq \mathrm{dom}(p)$ such that $(\underline{p}, \overline{A}) \models \varphi$. This means that the $+\omega$-picture $q = (p, \overline{A})$ belongs to L''. Let $\gamma = \gamma_{\hat{q}}$ be the type of $\hat{q}$. We know that L'' is a union of equivalence classes of $\sim_{d,t}$. Thus, every $+\omega$-picture r over Γ where $\hat{r}$ has the same type as $\hat{q}$ (i.e., $\gamma_{\hat{r}} = \gamma_{\hat{q}}$)

also belongs to L''. Let $j_0 \in \mathbb{N}_{\geqslant 1}$ be sufficiently large such that for all columns $j \geqslant j_0$ the $+\omega$-picture $\hat{q}$ has the same type as the finite picture $\hat{q}[1,j]$. Let j_1 be the first position $> j_0$ of p that carries the letter a. By our choice of n, the ω-word p has letter b at all the positions from j_1+1 to $j_1+1+(2^{kd}+1)\cdot d$. Let $J = \{j_1+1+id : i \in [2^{kd}+1]\}$. Consider all positions of p that belong to J. By the pigeon hole principle, there exist two positions $j, j' \in J$ with $j < j'$ such that $q[j-d+1, j] = q[j'-d+1, j']$. Now let r be the $+\omega$-picture defined by

$$r \quad = \quad q[1,j] \ \circledcirc \ \big(q[j+1,j']\big)^{\circledcirc\omega} \ .$$

It is straightforward to verify that $\hat{r}$ has the same type as $\hat{q}$ (i.e., $\gamma_{\hat{r}} = \gamma_{\hat{q}}$). Therefore, $r \in L''$. Let s be the ω-word over Σ and let $\overline{B} = B_1, \ldots, B_k \subseteq \mathrm{dom}(r)$ such that $r = (s, \overline{B})$. Since $r \in L''$, we know that $(\underline{s}, \overline{B}) \models \varphi$. Therefore, $\underline{s} \models \Phi$, and hence $s \in L'$. However, it is straightforward to see that our choice of j and j' implies that s contains only *finitely* many occurrences of the letter a. A contradiction! This completes the proof of Proposition 2.13. □

2.6 No Büchi Characterization Theorem for $+\omega$-Picture Languages

As we mentioned before, the well-known Büchi characterization theorem for ω-regular languages does not carry over to the Büchi-tiling systems. By Lemma 2.10 and Proposition 2.9 we can easlily show that: If L is a $+\omega$-picture language that is the union of a finite number of sets of the form $L_1 \circledcirc L_2^{\circledcirc\omega}$, where $L_1, L_2 \subseteq \Sigma^{++}$ are tiling recognizable sets of finite pictures, then L is Büchi-tiling recognizable. In this section, using combinatorial arguments we will show that the opposit direction is not true, even if we drop the requirement that L_1 and L_2 are tiling recognizable.

Theorem 2.28. *Let Σ be an alphabet with $|\Sigma| \geqslant 2$. For every $m \in \mathbb{N}_{\geqslant 1}$ let L_m be the language consisting of all $+\omega$-pictures over Σ that are of the form $s_1 \circledcirc s_2 \circledcirc s_3 \circledcirc \cdots$ where $s_\nu \in \Sigma^{m,m}$ for every $\nu \in \mathbb{N}_{\geqslant 1}$ and $s_\nu \neq s_1$ for infinitely many $\nu \in \mathbb{N}_{\geqslant 1}$. Then, the $+\omega$-picture language $L = \bigcup_{m \in \mathbb{N}_{\geqslant 1}} L_m$ is Büchi-tiling recognizable, but not equal to any union of a finite number of sets of the form $L_1 \circledcirc L_2^{\circledcirc\omega}$ with $L_1, L_2 \subseteq \Sigma^{++}$.*

Proof. For proving the first statement, we show that L is $\mathrm{EMSO}^{\infty}[\tau_\Sigma]$-definable and then use Theorem 2.14. The essential idea for constructing the $\mathrm{EMSO}^{\infty}[\tau_\Sigma]$-formula is to "guess" a position $z = (i,j)$ of s_1 such that for infinitely many ν the letter of s_ν at position (i,j) is different from the

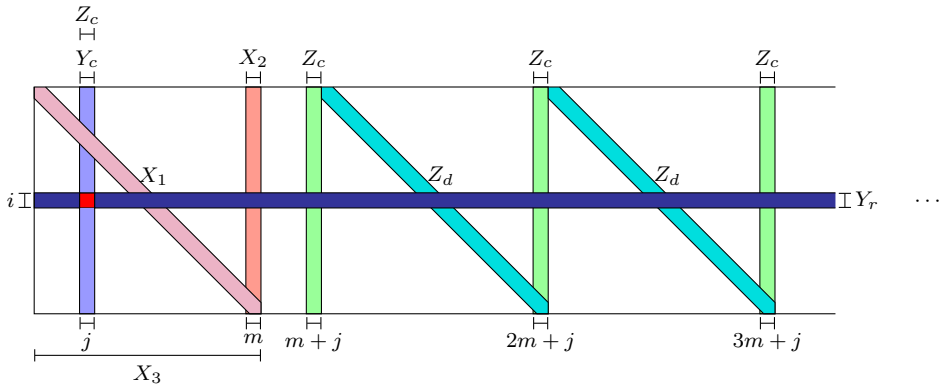


Figure 2.1: A picture indicating the sets X_1, X_2, X_3, Y_r, Y_c, Z_c and Z_d

letter of s_1 at position (i, j). To do this, we use a quantifier $\exists^\infty Z$ for the set of positions (i, j) in s_ν for the suitable ν. To make sure that z indeed belongs to s_1, we use further existential quantifiers X_1, X_2, X_3. X_1 consists of all positions in the diagonal of s_1, i.e., the diagonal starting at the topleft position of the picture and proceeding from one position to the one in the next row and next column, X_2 consists of all positions in the rightmost column of s_1, X_3 consists of all positions of s_1. To make sure that the set Z only contains positions in row i and in columns of the form $m{\cdot}k + j$, for $k \geqslant 1$, we use additional existential quantifiers Y_r, Y_c, Z_c, Z_d with the intended meaning that Y_r consists of all positions in the same row as z (i.e., row i), Y_c consists of all positions in the same column as z (i.e., column j), Z_c consists of all positions in column j, column $j{+}m$, column $j{+}2m$, and in general, column $j{+}km$ for all $k \geqslant 0$ (m denotes the height of the $+\omega$-picture in which the formula is evaluated), Z_d consists of all positions in the diagonals that start at positions in the top row which are directly to the right of positions in Z_c (i.e., positions of the form $(1, km + j{+}1)$, for $k \geqslant 0$) and that proceed from one position to the one in the next row and next colum. Note that each diagonal in Z_d ends in the bottom row at position $\big(m, (k{+}1)m{+}j\big)$, for $k \geqslant 0$. Figure 2.1 indicates the sets X_1, X_2, X_3, Y_r, Y_c, Z_c and Z_d over a picture of the form $s_1 \oplus s_2 \oplus s_3 \oplus \cdots$ where $s_\nu \in \Sigma^{m,m}$ for every $\nu \in \mathbb{N}_{\geqslant 1}$ and $s_\nu \neq s_1$ for infinitely many $\nu \in \mathbb{N}_{\geqslant 1}$. Z is then required to be an infinite set of positions $z' \in Y_r \cap Z_c$ such the letter at position z' is different from the letter at position z. The EMSO$^\infty[\tau_\Sigma]$-sentence that defines L is the sentence

$$\Phi \quad = \quad \exists^\infty Z\ \exists X_1\ \exists X_2\ \exists X_3\ \exists Y_r\ \exists Y_c\ \exists Z_c\ \exists Z_d\ \exists z\ \varphi$$

where φ is a FO[τ_Σ]-formula of the form

$$\bigwedge_{i=1}^{7} \varphi_i \quad \wedge \quad z \in X_3$$
$$\wedge \quad \forall z' \Big(z' \in Z \rightarrow \big(\, z' \in Y_r \wedge z' \in Z_c \wedge \bigwedge_{a \in \Sigma} \neg (P_a(z) \wedge P_a(z')) \, \big) \Big),$$

and each φ_i, which will be given below, is a FO[τ_Σ]-formula defined in such a way that the intended meaning of the existential quantifiers X_1, X_2, X_3, Y_r, Y_c, Z_c and Z_d will be satisfied. For formulating the φ_i, the following notation will be convenient: For two positions x and y of a picture, we say that y is the *diagonal successor* of x (and x is the *diagonal predecessor* of y) if y is reached from x by going to the next row and the next column. The formula $\varphi_{ds}(x,y) = \exists v \, \big(x S_h v \wedge v S_v y \big)$ expresses that y is the diagonal successor of x. We choose the formulas $\varphi_1, \ldots, \varphi_7$ as follows:

1. φ_1 states that X_1 is closed under diagonal successors and diagonal predecessors, and that the position $x = (1,1)$ in the upper left corner of the picture is the only element of X_1 that is in the top row or the leftmost column of the picture. Thus, we choose φ_1 as follows:

$$\forall x \forall y \, \Big(\, \varphi_{ds}(x,y) \; \rightarrow \; \big(x \in X_1 \leftrightarrow y \in X_1 \big) \, \Big)$$
$$\wedge \quad \forall x \, \Big(\, \big(\varphi_l(x) \vee \varphi_t(x) \big) \; \rightarrow \; \big(\, x \in X_1 \leftrightarrow \varphi_{tl}(x) \, \big) \, \Big).$$

2. φ_2 states that X_2 is closed under vertical successors and vertical predecessors, and the only element of X_2 that is in the bottom row is the bottom-row position that belongs to X_1. Thus, we choose φ_2 as follows:

$$\forall x \forall y \, \Big(x S_h y \rightarrow \big(x \in X_2 \leftrightarrow \; y \in X_2 \big) \Big)$$
$$\wedge \, \forall x \, \Big(\varphi_b(x) \; \rightarrow \; \big(x \in X_2 \leftrightarrow x \in X_1 \big) \Big).$$

3. φ_3 states that X_3 is closed under horizontal predecessors and contains all elements of X_2, but no element that is a horizontal successor of an element in X_2. Thus, we choose φ_3 as follows:

$$\forall x \forall y \, \Big(\big(x S_v y \wedge y \in X_3 \big) \rightarrow x \in X_3 \Big)$$
$$\wedge \quad \forall x \, \Big(x \in X_2 \rightarrow x \in X_3 \Big)$$
$$\wedge \, \forall x \forall y \, \Big(\big(x \in X_2 \wedge x S_v y \big) \rightarrow \neg (y \in X_3) \Big).$$

4. φ_4 states that Y_r contains z, is closed under horizontal successors and horizontal predecessors, and contains only one position in the left border of the picture. Thus, we choose φ_4 as follows:

$$\begin{aligned} z \in Y_r \quad \wedge \quad & \forall x \forall y \Big(\, x S_v y \; \rightarrow \; \big(\, x \in Y_r \leftrightarrow y \in Y_r \, \big) \, \Big) \\ & \wedge \forall x \, \forall y \, \Big(\, \big(\, \varphi_l(x) \; \wedge \; \varphi_l(y) \; \wedge \; x \in Y_r \; \wedge \; y \in Y_r \, \big) \; \rightarrow \; x = y \Big). \end{aligned}$$

5. Accordingly, φ_5 states that Y_c contains z, is closed under vertical successors and vertical predecessors, and contains only one position in the top row of the picture. Thus, we choose φ_5 as follows:

$$\begin{aligned} z \in Y_c \quad \wedge \quad & \forall x \forall y \Big(\, x S_h y \; \rightarrow \; \big(\, x \in Y_c \leftrightarrow y \in Y_c \, \big) \, \Big) \\ & \wedge \forall x \, \forall y \, \Big(\, \big(\, \varphi_t(x) \; \wedge \; \varphi_t(y) \; \wedge \; x \in Y_c \; \wedge \; y \in Y_c \, \big) \; \rightarrow \; x = y \Big). \end{aligned}$$

6. φ_6 states that Z_c contains Y_c and no further element of X_3, Z_c is closed under vertical successors and vertical predecessors, and the bottom-row elements of Z_c that do not belong to X_3 are exactly the bottom-row elements that belong to Z_d. Thus, we choose φ_6 as follows:

$$\begin{aligned} \forall x \, \Big(x \in X_3 \rightarrow \big(x \in Z_c \leftrightarrow x \in Y_c \big) \Big) & \\ \wedge \forall x \forall y \, \Big(x S_h y \rightarrow \big(x \in Z_c \leftrightarrow y \in Z_c \big) \Big) & \\ \wedge \forall x \, \Big(\big(\varphi_b(x) \wedge \neg (x \in X_3) \big) \rightarrow \big(x \in Z_c \leftrightarrow x \in Z_d \big) \Big). & \end{aligned}$$

7. φ_7 states that Z_d is closed under diagonal successors and diagonal predecessors, and the top-row elements of Z_d are exactly the horizontal successors of the top-row elements of Z_c. Thus, we choose φ_7 as follows:

$$\begin{aligned} \forall x \forall y \, \Big(\varphi_{ds}(x,y) \rightarrow \big(x \in Z_d \leftrightarrow y \in Z_d \big) \Big) & \\ \wedge \forall x \, \Big(\varphi_t(x) \rightarrow \big(x \in Z_d \leftrightarrow \exists y \, (y \in Z_c \wedge y S_v x) \big) \Big). & \end{aligned}$$

It is straightforward (but somewhat tedious) to check that the formula Φ indeed has the intended meaning and therefore defines the $+\omega$-picture language L.

For proving the second statement, by contradiction, assume that there is a $k \in \mathbb{N}_{\geqslant 1}$ and there are languages $L_{\kappa 1}, L_{\kappa 2} \subseteq \Sigma^{++}$ for $\kappa \in [k]$ such that

$$L \quad = \quad \bigcup_{\kappa=1}^{k} \, \big(\, L_{\kappa 1} \oplus L_{\kappa 2}^{\oplus \omega} \, \big).$$

Claim 2.29. *Let $m \in \mathbb{N}_{\geqslant 1}$ and let $s, t \in \Sigma^{m,m}$ with $s \neq t$. There exist numbers $\kappa \in [k]$ and $r \in [m]$ such that $\big(t[r,m] \oplus t^{\oplus\omega}\big) \in L_{\kappa 2}^{\oplus\omega}$. Furthermore, there exist numbers $\ell, n \in \mathbb{N}_{\geqslant 1}$ and a finite picture u of height m such that the picture $(s \oplus u)$ belongs to $L_{\kappa 1} \oplus L_{\kappa 2}^{\oplus \ell}$ and has width $m \cdot n + (r-1)$.*

Proof. By definition of L, the $+\omega$-picture $p = (s \oplus t^{\oplus\omega})$ belongs to L. Thus, there exists a $\kappa \in [k]$ such that $p \in L_{\kappa 1} \oplus L_{\kappa 2}^{\oplus\omega}$. I.e., there is an infinite sequence of numbers $1 < i_1 < i_2 < i_3 < \cdots$ such that $p[1, i_1{-}1] \in L_{\kappa 1}$ and $p[i_\nu, i_{\nu+1}{-}1] \in L_{\kappa 2}$ for every $\nu \in \mathbb{N}_{\geqslant 1}$. Now let $j_1 < j_2 < j_3 < \cdots$ be an infinite subsequence of the sequence $i_1 < i_2 < i_3 < \cdots$ (i.e., $\{j_1, j_2, j_3, \cdots\} \subseteq \{i_1, i_2, i_3, \cdots\}$) for which the following is true: $j_1 > \max\{2m, i_2\}$ and $j_{\mu+1} > j_\mu + m$ for every $\mu \in \mathbb{N}_{\geqslant 1}$. For each j_μ let $n_\mu \in \mathbb{N}$ and $r_\mu \in \{1, \ldots, m\}$ be such that $j_\mu = n_\mu \cdot m + r_\mu$. By the pigeon hole principle, there must exist $\mu, \mu' \in \{1, \ldots, m{+}1\}$ with $\mu < \mu'$ and $r_\mu = r_{\mu'}$. Now, consider the picture $q = p[j_\mu, j_{\mu'}{-}1]$. It is straightforward to see that if $r_\mu = 1$ then q is of the form $t^{\oplus c}$ for some $c \in \mathbb{N}_{\geqslant 1}$. Similarly, if $r_\mu \neq 1$, then q is of the form $t[r_\mu, m] \oplus t^{\oplus c} \oplus t[1, r_\mu{-}1]$ for some $c \in \mathbb{N}_{\geqslant 1}$. By our choice of the sequence $j_1 < j_2 < \cdots$ we know that $q \in L_{\kappa 2}^{\oplus d}$ for some $d \in \mathbb{N}_{\geqslant 1}$. Thus, $q^{\oplus\omega} \in L_{\kappa 2}^{\oplus\omega}$. Furthermore, if $r_\mu = 1$ then $q^{\oplus\omega} = t^{\oplus\omega} = (t[1,m] \oplus t^{\oplus\omega})$, and otherwise $q^{\oplus\omega} = (t[r_\mu, m] \oplus t^{\oplus\omega})$. Thus, for the chosen κ and for $r = r_\mu$ we have shown that $(t[r,m] \oplus t^{\oplus\omega}) \in L_{\kappa 2}^{\oplus\omega}$. We also know that the picture $p[1, j_\mu{-}1]$ belongs to $L_{\kappa 1} \oplus L_{\kappa 2}^{\oplus \ell}$ for some $\ell \in \mathbb{N}_{\geqslant 1}$, is of width $j_\mu{-}1 = n_\mu \cdot m + (r_\mu - 1)$, and is of the form $(s \oplus u)$ for the picture $u = p[m{+}1, j_\mu{-}1]$. This completes the proof of Claim 2.29. □

For each picture $t \in \Sigma^{m,m}$ let $K(t)$ be the set of all $(\kappa, r) \in [k] \times [m]$ such that $(t[r,m] \oplus t^{\oplus\omega}) \in L_{\kappa 2}^{\oplus\omega}$. Claim 2.29 implies for all $t \in \Sigma^{m,m}$ that $K(t) \neq \emptyset$ (note that here we use that $|\Sigma| \geqslant 2$). Let us now choose a number $m \in \mathbb{N}_{\geqslant 1}$ such that

$$|\Sigma|^{(m^2)} \;>\; 2^{k \cdot m} \tag{2.2}$$

(note that this is true for all sufficiently large m, since k is fixed, $|\Sigma| \geqslant 2$, and the function $2^{(x^2)}$ grows faster than the function 2^{kx}). From equation (2.2) we obtain pictures $t_1, t_2 \in \Sigma^{m,m}$ with $t_1 \neq t_2$ and $K(t_1) = K(t_2)$. Now apply Claim 2.29 for $s = t_1$ and $t = t_2$. This yields numbers $(\kappa, r) \in K(t_2)$, a finite picture u of height m, and numbers $\ell, n \in \mathbb{N}_{\geqslant 1}$ such that the picture $(t_1 \oplus u)$ belongs to $L_{\kappa 1} \oplus L_{\kappa 2}^{\oplus \ell}$ and has width $m \cdot n + (r-1)$. Since $(\kappa, r) \in K(t_2) = K(t_1)$, we know that $(t_1[r,m] \oplus t_1^{\oplus\omega}) \in L_{\kappa 2}^{\oplus\omega}$. Thus, the $+\omega$-picture

$$p \;=\; (t_1 \oplus u) \oplus t_1[r,m] \oplus t_1^{\oplus\omega}.$$

belongs to $L_{\kappa 1} \oplus L_{\kappa 2}^{\oplus \omega}$. Since $(t_1 \oplus u \oplus t_1[r, m])$ has width $(m{+}1) \cdot n$, the picture p obviously does *not* belong to the language L. A contradiction! This completes the proof of Theorem 2.28. □

Chapter 3

Weighted Register Automata and Weighted Logic on Data Words

Data words are sequences of pairs where the first element is taken from a finite alphabet (as in classical words) and the second element is taken from an infinite data domain. *Register automata*, introduced by Kaminski and Francez [71], provide a widely studied model for reasoning on data words. These automata can be considered as classical nondeterministic finite automata equipped with a finite set of registers which are used to store data in order to compare them with some data in the future. In this chapter, for quantitative reasoning on data words, we introduce *weighted register automata* over *commutative data semirings* equipped with a collection of binary data functions in the spirit of the classical theory of weighted automata [42]. Whereas in the models of register automata known from the literature data are usually compared with respect to equality or a linear order, here we allow data comparison by means of an arbitrary collection of binary data relations. This approach permits easily to incorporate timed automata [2] and weighted timed automata [5, 86] into our framework. Motivated by the seminal Büchi-Elgot-Trakhtenbrot theorem [25] about the expressive equivalence of finite automata and monadic second-order (MSO) logic and by the weighted MSO logic of Droste and Gastin [41], we introduce *weighted MSO logic on data words* and give a logical characterization of weighted register automata.

3.1 Register Automata

Register automata are nondeterministic finite state automata on data words equipped with a finite set of registers for storing data (from an infinite data domain) and comparing new data instances with the already stored data. Whereas in the original definition of register automata [71] it is only possible to check the equality of data, the later models augment a data domain with a linear order and allow to compare data with respect to this linear order. However, more complicated situations of data comparison are reasonable as well (cf., e.g., timed automata [2] or Example 3.2). Here, we consider the model of register automata where we augment a data domain with some arbitrary binary relations which will be used for some more general data comparison.

For a set X, let $\mathcal{P}(X) = \{Y \mid Y \subseteq X\}$, the *power set* of X. A *data structure* is a pair $\mathbb{D} = \langle \mathsf{D}, \mathcal{R} \rangle$ where D is an arbitrary set called a *data domain* and $\mathcal{R}$ is a set of binary relations on D called *data relations*. For the convenience of presentation, we assume throughout all of this chapter that D contains a designated initial data value $\perp \in \mathsf{D}$ which will be the initial data value of all registers. Let Σ be an alphabet and $\mathbb{D} = \langle \mathsf{D}, \mathcal{R} \rangle$ a data structure. A *data word* over Σ and $\mathbb{D}$ is a finite sequence $w = (a_1, d_1)...(a_n, d_n)$ where $a_1, ..., a_n \in \Sigma$ and $d_1, ..., d_n \in \mathsf{D}$. Let $\mathbb{D}\Sigma^+ = (\Sigma \times \mathsf{D})^+$ denote the set of all data words over Σ and $\mathbb{D}$. Any subset $L \subseteq \mathbb{D}\Sigma^+$ is called a *data language*.

Let Reg be a finite set of *registers* which take values from the data domain D of the data structure $\mathbb{D}$. A *register valuation* over Reg and $\mathbb{D}$ is any mapping $\nu : \mathsf{Reg} \to \mathsf{D}$; by a slight abuse of notation, we can consider a register valuation ν as a vector $\nu = \begin{pmatrix} \nu(r_1) \\ \cdots \\ \nu(r_m) \end{pmatrix} \in \mathsf{D}^m$ where $\mathsf{Reg} = \{r_1, \cdots, r_m\}$ is an enumeration on the registers $r_i \in \mathsf{Reg}$, $1 \leqslant i \leqslant m$. For a register valuation $\nu \in \mathsf{D}^{|\mathsf{Reg}|}$, a subset $\mathsf{up} \subseteq \mathsf{Reg}$ and a data value $d \in \mathsf{D}$, the *update* $\nu[\mathsf{up} := d]$ is the register valuation in $\mathsf{D}^{|\mathsf{Reg}|}$ defined for all registers r by $\nu[\mathsf{up} := d](r) = d$ if $r \in \mathsf{up}$ and $\nu[\mathsf{up} := d](r) = \nu(r)$ otherwise.

The set $\Phi(\mathsf{Reg}, \mathbb{D})$ of *register guards* over Reg and $\mathbb{D}$ is defined by the grammar

$$\varphi \ ::= \mathsf{True} \mid Rr \mid \varphi \wedge \varphi \mid \neg\varphi$$

where $r \in \mathsf{Reg}$ and R is a data relation in D. Given a register valuation $\nu \in \mathsf{D}^{|\mathsf{Reg}|}$, a data value $d \in \mathsf{D}$ and a register guard $\varphi \in \Phi(\mathsf{Reg}, \mathbb{D})$, the *satisfaction relation* $(\nu, d) \models \varphi$ is defined inductively on the structure of φ as follows: $(\nu, d) \models \mathsf{True}$ always holds; $(\nu, d) \models Rr$ iff $(\nu(r), d) \in R$;

$(\nu, d) \models \varphi_1 \wedge \varphi_2$ iff $(\nu, d) \models \varphi_1$ and $(\nu, d) \models \varphi_2$; $(\nu, d) \models \neg\varphi$ iff $(\nu, d) \models \varphi$ does not hold.

Definition 3.1. *Let Σ be an alphabet and $\mathbb{D}$ a data structure. A* register automaton*, for short* RA*, over Σ and $\mathbb{D}$ is a tuple $\mathcal{A} = (\mathcal{L}, \mathsf{Reg}, \mathcal{L}_i, T, \mathcal{L}_f)$ where $\mathcal{L}$ is a finite set of* locations*, Reg is a finite set of* registers*, $\mathcal{L}_i, \mathcal{L}_f \subseteq \mathcal{L}$ are sets of* initial *resp.* final *locations, and $T \subseteq \mathcal{L} \times \Sigma \times \Phi(\mathsf{Reg}, \mathbb{D}) \times 2^{\mathsf{Reg}} \times \mathcal{L}$ is a finite set of* transitions*.*

We will denote a transition $t = (\ell, a, \varphi, \mathsf{up}, \ell') \in T$ by $\ell \xrightarrow{\varphi,\, a,\, \mathsf{up}} \ell'$. We may omit φ when φ is True, and omit up if $\mathsf{up} = \emptyset$. We write $\downarrow r$ when $\mathsf{up} = \{r\}$. Let $\mathrm{label}(t) = a \in \Sigma$, the *label* of t. A *state* of $\mathcal{A}$ is a pair $q = \langle \ell, \nu \rangle \in \mathcal{L} \times \mathsf{D}^{|\mathsf{Reg}|}$ consisting of a location $\ell \in \mathcal{L}$ and a register valuation $\nu \in \mathsf{D}^{|\mathsf{Reg}|}$. We say that q is *initial* if $\ell \in \mathcal{L}_i$ and $\nu(r) = \perp$ for all $r \in \mathsf{Reg}$. We call q *final* if $\ell \in \mathcal{L}_f$. Let $q = \langle \ell, \nu \rangle$ and $q' = \langle \ell', \nu' \rangle$ be states of $\mathcal{A}$, $t \in T$ a transition of the form $\hat{\ell} \xrightarrow{\varphi,\, a,\, \mathsf{up}} \hat{\ell}'$, and $d \in \mathsf{D}$ a data value. We say that $q \vdash_{t,d} q'$ is a *switch* from q to q' via the transition t and the data value d if $\hat{\ell} = \ell$, $\hat{\ell}' = \ell'$, $(\nu, d) \models \varphi$ and $\nu' = \nu[\mathsf{up} := d]$. Note that q' is uniquely determined by q, t and d. A *run* ρ of $\mathcal{A}$ is a non-empty sequence of switches between states starting in an initial state and ending in a final state. Formally, ρ is a sequence of the form $q_0 \vdash_{t_1,d_1} q_1 \vdash_{t_2,d_2} \ldots \vdash_{t_n,d_n} q_n$ where $n \geq 1$, q_0 is an initial state, $q_1, \ldots, q_{n-1}$ are states, q_n is a final state, $t_1, \ldots, t_n \in T$ and $d_1, \ldots, d_n \in \mathsf{D}$. Let $\mathrm{label}(\rho) = (\mathrm{label}(t_1), d_1) \ldots (\mathrm{label}(t_n), d_n) \in \mathbb{D}\Sigma^+$, the *label* of ρ. For any data word $w \in \mathbb{D}\Sigma^+$, let $\mathrm{Run}_{\mathcal{A}}(w)$ denote the set of all runs of $\mathcal{A}$ with label w. Let $L(\mathcal{A}) = \{w \in \mathbb{D}\Sigma^+ \mid \mathrm{Run}_{\mathcal{A}}(w) \neq \emptyset\}$, the data language *recognized* by $\mathcal{A}$.

Example 3.2. *Now we give some examples of data structures for register automata. For any data domain D, let $(=_{\mathsf{D}}) \subseteq \mathsf{D} \times \mathsf{D}$ denote the data relation $\{(d, d) \mid d \in \mathsf{D}\}$.*

1. Let D be any non-empty set. Then, $\mathbb{D} = (\mathsf{D}, \{=_{\mathsf{D}}\})$ is a data structure which corresponds to the original model of register automata [71]. Note that the register automata of [71] are also equipped with an initial vector of data values. In order to model this feature in our setting, we can extend the set of data relations of $\mathbb{D}$ with the set $\{R_d \mid d \in \mathsf{D}\}$ where $R_d = \{(d', d) \mid d' \in \mathsf{D}\}$ for all $d \in \mathsf{D}$.

2. Let $(\mathsf{D}, <)$ be a linear order with a non-empty set D. Then, $(\mathsf{D}, \{=_{\mathsf{D}}, <_{\mathsf{D}}\})$ is a data structure for the register automata considered in [58].

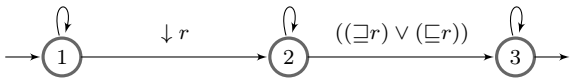


Figure 3.1: The register automaton $\mathcal{A}$ of Example 3.3

3. Data are often represented in the form of finite strings. Here we consider the model of register automata which permits to check whether one data string is contained in another. Let Γ *be an alphabet and* $\mathsf{D} = \Gamma^*$ *be the data domain. Let* $u, v \in \Gamma^*$*. We say that* u *is* contained *in* v*, written* $u \sqsubseteq v$*, if* $v = \alpha u \beta$ *with* $\alpha, \beta \in \Gamma^*$*. Based on this definition, we can define the data relations* $\sqsubseteq$ *and* $\sqsupseteq$ *on* D*. Then,* $(\mathsf{D}, \{\sqsubseteq, \sqsupseteq\})$ *is a data structure.*

4. In various situations, exact data values are not known and we deal with their approximated values, e.g., obtained from some experiments. Therefore, it can be reasonable to compare data values with respect to a given approximation error. For instance, let the data domain D *be the set of all rational numbers. For any nonnegative rational number* ε*, let* $R_\varepsilon = \{(d, d') \mid d, d' \in \mathsf{D} \text{ and } |d - d'| \leq \varepsilon\}$*. Note that* R_0 *is equal to* $=_{\mathsf{D}}$*. Then,* $(\mathsf{D}, \{R_\varepsilon \mid \varepsilon \geq 0\})$ *is a data structure for register automata with approximated values.*

Example 3.3. *Consider the data structure* $\mathbb{D} = (\mathsf{D}, \{\sqsubseteq, \sqsupseteq\})$ *of Example 3.2 (3) with* $\mathsf{D} = \Gamma^*$*. Let* $\Sigma = \{a\}$ *be a singleton alphabet. We say that that a word* $w = (a, d_1)...(a, d_n) \in \mathbb{D}\Sigma^+$ *is* redundant *if* $d_i \sqsubseteq d_j$ *for some* $i, j \in \{1, .., n\}$ *with* $i \neq j$*. Let* $L \subseteq \mathbb{D}\Sigma^+$ *be the data language of all redundant data words. The data language* L *is recognized by an* RA $\mathcal{A}$ *over* $\mathbb{D}$ *with one register* r *depicted in Fig.3.1. Here, we omit the transition label* a*, register guard* True *and the empty register update;*

3.2 Relation to Timed Automata

We show that *timed automata* [2] are also included in our model of register automata. Recall that timed automata are nondeterministic finite automata equipped with a finite set of clocks ranging over $\mathbb{R}_{\geq 0}$ and measuring the time of stay in the locations. The transitions of a timed automaton are also augmented with a constraint on clock values and a subset of clocks which must be reset after taking this transition.

Now we turn to a formal definition of a timed automaton (TA). Recall that, given a finite set of clocks C, a *clock constraint* over C is generated

by the grammar

$$\varphi \ ::= \mathsf{True} \mid x \bowtie k \mid \varphi \wedge \varphi \mid \neg\varphi$$

where $x \in C$ and $k \in \mathbb{N}$. A *clock valuation* over C is a mapping $\nu : C \to \mathbb{R}_{\geq 0}$. The definition that ν *satisfies* a clock constraint φ, written $\nu \models \varphi$, is given by induction on the structure of φ as usual. Given a clock valuation $\nu \in \mathbb{R}_{\geq 0}^C$, a delay $d \in \mathbb{R}_{\geq 0}$ and a set $\Lambda \subseteq C$ of clocks to be reset, the clock valuations $\nu + d \in \mathbb{R}_{\geq 0}^C$ and $\nu[\Lambda := 0] \in \mathbb{R}_{\geq 0}^C$ are defined for all clocks $x \in C$ by $(\nu + d)(x) = \nu(x) + d$, $\nu[\Lambda := 0](x)$ is equal to 0 if $x \in \Lambda$ and is $\nu(x)$ otherwise.

Definition 3.4. *A* timed automaton*, for short* TA*, over an alphabet* Σ *is a tuple* $\mathcal{A} = (\mathcal{L}, C, \mathcal{L}_i, E, \mathcal{L}_f)$ *where* $\mathcal{L}$ *is a finite set of locations,* $\mathcal{L}_i, \mathcal{L}_f \subseteq \mathcal{L}$ *are sets of initial resp. final locations,* C *is a finite set of clocks and* T *is a finite set of transitions* t *of the form* $\ell \xrightarrow{\varphi,\, a,\, \Lambda} \ell'$ *where* $\ell, \ell' \in \mathcal{L}$*,* $a \in \Sigma$*,* φ *is a clock constraint over* C *and* $\Lambda \subseteq C$ *is a set of clocks to be reset.*

Let $\mathrm{label}(t) = a$, the *label* of t. A *state* of $\mathcal{A}$ is a pair $q = \langle \ell, \nu \rangle \in \mathcal{L} \times \mathbb{R}_{\geq 0}^C$. Given states $q = \langle \ell, \nu \rangle$, $q' = \langle \ell', \nu' \rangle$ of $\mathcal{A}$, a transition t of $\mathcal{A}$ of the form $\hat{\ell} \xrightarrow{\varphi,\, a,\, \Lambda} \hat{\ell}'$ and a time delay $d \in \mathbb{R}_{\geq 0}$, we say that $q \vdash_{t,d} q'$ is a *switch* from q to q' after the time delay d via t if $\hat{\ell} = \ell$, $\hat{\ell}' = \ell'$, $(\nu + d) \models \varphi$ and $\nu' = (\nu + d)[\Lambda := 0]$. Then, as for register automata, we define a run ρ of $\mathcal{A}$ as a non-empty sequence of switches between states of the form $q_0 \vdash_{t_1,d_1} q_1 \vdash_{t_2,d_2} \ldots \vdash_{t_n,d_n} q_n$ where $q_0 \in \mathcal{L}_i \times \{0\}^C$ and $q_n \in \mathcal{L}_f \times \mathbb{R}_{\geq 0}^C$. Then, the *label* of ρ is the word $\mathrm{label}(\rho) = (\mathrm{label}(t_1), d_1)(\mathrm{label}(t_2), d_1 + d_2)\ldots(\mathrm{label}(t_n), \sum_{i=1}^n d_i)$ in $(\Sigma \times \mathbb{R}_{\geq 0})^+$.

A *timed word* over Σ is a word $w = (a_1, \tau_1)\ldots(a_n, \tau_n) \in (\Sigma \times \mathbb{R}_{\geq 0})^+$ such that $\tau_1 \leq \tau_2 \leq \ldots \leq \tau_n$. Let $\mathbb{T}\Sigma^+$ denote that set of all timed words over Σ. Note that, given a TA and a run of it, the label of this run is necessarily a timed word. Any set of timed words is called a *timed language*. The timed language *recognized* by $\mathcal{A}$ is defined to be the set of all timed words w over Σ such that there exists a run ρ of $\mathcal{A}$ with $\mathrm{label}(\rho) = w$.

In order to simulate timed automata by register automata, we consider the data structure $\mathbb{D}^{\mathrm{Timed}} = (\mathsf{D}, \{R_{\bowtie k} \mid \bowtie \in \{<, =, >\} \text{ and } k \in \mathbb{N}\})$ where $\mathsf{D} = \mathbb{R}_{\geq 0}$ with the initial data value $\perp = 0$ and, for $\bowtie \in \{<, =, >\}$ and $k \in \mathbb{N}$, $R_{\bowtie k} = \{(\tau, \tau') \mid \tau, \tau' \in \mathsf{D} \text{ and } \tau' - \tau \bowtie k\}$. Note that $R_{=0}$ is equal to $=_D$.

Lemma 3.5. *Let Σ be an alphabet. Then every recognizable timed language over Σ is recognizable by an* RA *over Σ and $\mathbb{D}^{\mathrm{Timed}}$.*

Proof. Let $\mathcal{A} = (\mathcal{L}, C, \mathcal{L}_i, T, \mathcal{L}_f)$ be a TA over Σ. We show that there exists a register automaton $\mathcal{A}^{\mathsf{data}}$ over Σ and $\mathbb{D}^{\mathrm{Timed}}$ such that $L(\mathcal{A}^{\mathsf{data}}) = L(\mathcal{A})$. We let $\mathcal{A}^{\mathsf{data}} = (\mathcal{L}, \mathsf{Reg}, \mathcal{L}_i, T', \mathcal{L}_f)$ where:

- $\mathsf{Reg} = C \cup \{\tilde{r}\}$ where the register $\tilde{r} \notin C$ will be used to check monotonicity of the timed part of a timed word,
- every transition $t \in T$ of the form $\ell \xrightarrow{\varphi,\, a,\, \Lambda} \ell'$ is simulated in $\mathcal{A}^{\mathsf{data}}$ by the transition $\Psi(t) = \big(\ell \xrightarrow{\varphi',\, a,\, \mathsf{up} \cup \{\tilde{r}\}} \ell'\big)$ where $\mathsf{up} = \Lambda$ and $\varphi' = \tilde{\varphi} \wedge (R_{\geq 0}\tilde{r})$ such that $\tilde{\varphi} \in \Phi(\mathsf{Reg}, \mathbb{D}^{\mathrm{Timed}})$ is obtained from φ by replacing every constraint $x \bowtie k$ by $(R_{\bowtie k})x$. In other words, we let $T' = \Psi(T)$.

This construction relies on the fact that the value of a clock at a time stamp τ is the difference between τ and the time stamp of the previous clock reset before τ. Then, whenever the timed automaton $\mathcal{A}$ resets a clock x, the register automaton $\mathcal{A}^{\mathsf{data}}$ stores in the register x the time stamp of this reset. Although negative time delays are admitted by register automata over $\mathbb{D}^{\mathrm{Timed}}$, they are not possible in $\mathcal{A}^{\mathsf{data}}$ since it checks using the register $\tilde{r}$ that all time delays are nonnegative. Then $L(\mathcal{A}^{\mathsf{data}}) = L(\mathcal{A})$ and hence the claim follows. □

Note that register automata over $\mathbb{D}^{\mathrm{Timed}}$ are more expressive than timed automata since they can accept non-monotonic data words. Nevertheless, the following holds true:

Lemma 3.6. *Let Σ be an alphabet and $L \subseteq \mathbb{T}\Sigma^+$. Then, L is recognizable by an* RA *over Σ and $\mathbb{D}^{\mathrm{Timed}}$ iff L is recognizable by a* TA *over Σ.*

Proof. The implication $\Leftarrow$ follows immediately from Lemma 3.5. We show the converse implication. Let $\mathcal{A}^{\mathsf{data}} = (\mathcal{L}, \mathsf{Reg}, \mathcal{L}_i, T, \mathcal{L}_f)$ be an RA over Σ and $\mathbb{D}^{\mathrm{Timed}}$ such that $L(\mathcal{A}^{\mathsf{data}}) = L$. Consider the timed automaton $\mathcal{A}^{\mathrm{time}} = (\mathcal{L}, C, \mathcal{L}_i, T', \mathcal{L}_f)$ over Σ where $C = \mathsf{Reg}$ and T' is defined as follows. Every transition $t = \big(\ell \xrightarrow{\varphi,\, a,\, \mathsf{up}} \ell'\big) \in T$ is simulated in $\mathcal{A}^{\mathrm{time}}$ by the transition $t' = \big(\ell \xrightarrow{\tilde{\varphi},\, a,\, \Lambda} \ell'\big) \in T'$ where $\Lambda = \mathsf{up}$ and the clock constraint $\tilde{\varphi}$ over C is obtained from φ by replacing every register guard $(R_{\bowtie k})x$ by the atomic clock constant $x \bowtie k$. Then, the fact that $L(\mathcal{A}^{\mathsf{data}}) \subseteq \mathbb{T}\Sigma^+$ guarantees that $L(\mathcal{A}^{\mathrm{time}}) = L(\mathcal{A}^{\mathsf{data}}) = L$. Hence the claim follows. □

3.3 Weighted Register Automata

In this section, we introduce *weighted register automata* as a quantitative model for reasoning about data words. They extend the qualitative register automata of the previous section with weights and reflect the following quantitative information: As in classical weighted automata [42], transitions of our new weighted register automata also carry weights which do not depend on data. In addition, as opposed to weighted automata, weighted register automata must be able to process data taken from an infinite data domain. Therefore, the size of data can be very large and processing of such data can be expensive. Our model of weighted register automata takes into account the costs of data processing, namely, the costs of storing data in registers and the cost of comparing a new datum with old data stored in registers.

In order to be able to reflect various quantitative settings, we will consider a general structure for weighted register automata. Thus, we adopt the structure of semirings (as in the classical weighted automata [42]) to our new setting of data words.

For any sets X, Y, let Y^X denote the collection of all mappings $f : X \to Y$. For $y \in Y$, let y^X denote the mapping $y^X : X \to \{y\}$. A *data semiring* over a data structure $\mathbb{D} = (\mathsf{D}, \mathcal{R})$ is a pair $\mathbb{S} = \langle \mathcal{S}, F \rangle$ where $\mathcal{S} = (S, +, \cdot, \mathbf{0}, \mathbf{1})$ is a semiring and $\mathcal{F} \subseteq S^{\mathsf{D}\times\mathsf{D}}$ such that $\mathbf{1}^{\mathsf{D}\times\mathsf{D}} \in \mathcal{F}$. Let $\mathrm{dom}(\mathbb{S}) = S$, the *domain* of $\mathbb{S}$. We call $\mathbb{S}$ *commutative* if $\mathcal{S}$ is a commutative semiring, i.e., $s \cdot s' = s' \cdot s$ for all $s, s' \in S$.

Definition 3.7. *Let Σ be an alphabet, $\mathbb{D}$ a data structure and $\mathbb{S} = \langle (S, +, \cdot, \mathbf{0}, \mathbf{1}), \mathcal{F} \rangle$ a data semiring over $\mathbb{D}$. A* weighted register automaton (wRA) *over Σ, $\mathbb{D}$ and $\mathbb{S}$ is a tuple $\mathcal{A} = (\mathcal{L}, \mathsf{Reg}, \mathcal{L}_i, T, \mathcal{L}_f, \mathrm{wt})$ where $(\mathcal{L}, \mathsf{Reg}, \mathcal{L}_i, T, \mathcal{L}_f)$ is an* RA *over Σ and $\mathbb{D}$, and $\mathrm{wt} = \langle \mathrm{wt}_{\mathrm{trans}}, \mathrm{wt}_{\mathsf{data}} \rangle$ is a pair of weight functions such that $\mathrm{wt}_{\mathrm{trans}} : T \to S$ and $\mathrm{wt}_{\mathsf{data}} : (T \times \mathsf{Reg}) \to \mathcal{F}$.*

Note that $\mathrm{wt}_{\mathrm{trans}} : T \to S$ can be considered as a weight function of the classical weighted automata [42] and describes data-independent costs for transitions. The weight function $\mathrm{wt}_{\mathsf{data}}$ assigns to every transition $t \in T$, every register $r \in \mathsf{Reg}$, every data value d stored in r and every new data value $d' \in \mathsf{D}$ the cost $\mathrm{wt}_{\mathsf{data}}(t, r)(d, d') \in S$ of data processing in the register r which includes the cost of comparison d with d' and, if necessary, the cost of storing d' in the register r.

We fix an enumeration $(r_j)_{1\leq j\leq m}$ of Reg. Let $q = \langle \ell, \nu\rangle$ and $q' = \langle \ell', \nu'\rangle$ be states of $\mathcal{A}$ and $q \vdash_{t,d} q'$ a switch between them. Then, its *weight* is defined as the product of the costs of data processing in the registers (defined by $\mathrm{wt}_{\mathsf{data}}$) and the transition cost of $\mathrm{wt}_{\mathrm{trans}}(t)$. Formally, we let

$$\mathrm{wt}(q \vdash_{t,d} q') = \prod_{j=1}^{m} \mathrm{wt}_{\mathsf{data}}(t, r_j)(\nu(r_j), d) \cdot \mathrm{wt}_{\mathrm{trans}}(t).$$

Now let $\rho = \big(q_0 \vdash_{t_1,d_1} q_1 \vdash_{t_2,d_2} \ldots \vdash_{t_n,d_n} q_n\big)$ be a run of $\mathcal{A}$. Then, the *weight* of ρ is defined as the product of the weights of all switches of ρ. Formally, we let

$$\mathrm{wt}(\rho) = \prod_{i=1}^{n} \mathrm{wt}(q_{i-1} \vdash_{t_i,d_i} q_i).$$

Then, the *behavior* of $\mathcal{A}$ is the mapping $[\![\mathcal{A}]\!] : \mathbb{D}\Sigma^+ \to S$ defined for all $w \in \mathbb{D}\Sigma^+$ by

$$[\![\mathcal{A}]\!](w) = \sum \big(\, \mathrm{wt}(\rho) \mid \rho \in \mathrm{Run}_{\mathcal{A}}(w)\big).$$

We will call any mapping $\mathbb{L} : \mathbb{D}\Sigma^+ \to S$ a *data series* over Σ, $\mathbb{D}$ and $\mathbb{S}$. Let $\mathbb{S}\langle\langle \mathbb{D}\Sigma^+ \rangle\rangle$ denote the collection of all data series over Σ, $\mathbb{D}$ and $\mathbb{S}$. We say that $\mathbb{L} \in \mathbb{S}\langle\langle \mathbb{D}\Sigma^+ \rangle\rangle$ is *recognizable* if there exists a wRA $\mathcal{A}$ over Σ, $\mathbb{D}$ and $\mathbb{S}$ such that $[\![\mathcal{A}]\!] = \mathbb{L}$.

Example 3.8. *1. Consider the Boolean semiring* $\mathbb{B} = (\{0,1\}, \vee, \wedge, \mathbf{0}, \mathbf{1})$ *where* $\mathbf{1} \vee \mathbf{1} = \mathbf{1}$. *Let* $\mathbb{D} = (\mathsf{D}, \mathcal{R})$ *be a data structure, and let* $\mathcal{F} = \{\mathbf{1}^{\mathsf{D}\times\mathsf{D}}\}$. *Then* $\langle \mathbb{B}, \mathcal{F}\rangle$ *is a data boolean semiring, and a data language* $L \subseteq \mathbb{D}\Sigma^+$ *is recognizable if and only if its characteristic function* $\mathbf{1}_L : \mathbb{D}\Sigma^+ \to \mathbb{B}$ *(defined by* $\mathbf{1}_L(w) = \mathbf{1}$ *iff* $w \in L$*) is a recognizable data series.*

2. Consider the arctic semiring $\mathrm{Arc} = (\mathbb{N} \cup \{-\infty\}, \max, +, -\infty, 0)$ *of natural numbers. Let* $\mathbb{D} = (\mathsf{D}, \mathcal{R})$ *be a data structure augmented with a* size function $\mathrm{size} : \mathsf{D} \to \mathbb{N}$ *(e.g., length of a data string or number of bits of an integer). Consider the following collections of functions* $\mathcal{F}_1$ *and* $\mathcal{F}_2$ *defined as follows.*

- *Let* $\mathcal{F}_1$ *be the collection of functions* $f_c : \mathsf{D} \times \mathsf{D} \to \mathbb{N}$ *with* $c \in \mathbb{N}$ *and* $f_c(d, d') = c \cdot \mathrm{size}(d')$ *for all* $d, d' \in \mathsf{D}$. *The collection* $\mathcal{F}_1$ *can be useful, e.g., for the cases where we need to estimate the costs of checking the equality of data or to update a register.*
- *Let* $\mathcal{F}_2$ *be the collection of functions* $g_{k,l} : \mathsf{D} \times \mathsf{D} \to \mathbb{N}$ *with* $k, l \in \mathbb{N}$ *and* $g_{k,l}(d, d') = k \cdot \mathrm{size}(d) + l \cdot \mathrm{size}(d')$ *for all* $d, d' \in \mathsf{D}$. *The collection*

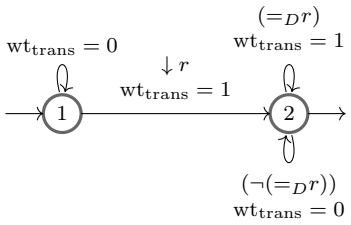


Figure 3.2: The wRA $\mathcal{A}$ of Example 3.9

> *$\mathcal{F}_2$ can be useful, e.g., for the cases where we need to estimate the costs of finding a pattern in a data string (e.g., using the well-known Knuth-Morris-Pratt algorithm).*

Then $\langle \mathrm{Arc}, \mathcal{F}_1 \rangle$ and $\langle \mathrm{Arc}, \mathcal{F}_2 \rangle$ are data semirings.

Example 3.9. *Consider the data structure $\langle \mathsf{D}, \{=_D\} \rangle$ of Example 3.2 (1) and a singleton alphabet $\Sigma = \{a\}$. For a data word $w \in \mathbb{D}\Sigma^+$ and $d \in \mathsf{D}$, let $|w|_d \in \mathbb{N}$ be the number of d's in w. Let $\mathbb{L} : \mathbb{D}\Sigma^+ \to \mathbb{N}$ be a data series defined by $\mathbb{L}(w) = \max_{d \in \mathsf{D}} |w|_d$. Consider the data semiring $\mathbb{S} = \langle \mathrm{Arc}, \{\mathbf{1}^{\mathsf{D}\times\mathsf{D}}\} \rangle$ and the* wRA $\mathcal{A}$ *over Σ, $\mathbb{D}$ and $\mathbb{S}$ depicted in Fig. 3.2 which has the only register r. Moreover, for all transitions t of $\mathcal{A}$, we put $\mathrm{wt}_{\mathsf{data}}(t, r) = \mathbf{1}^{\mathsf{D}\times\mathsf{D}}$. Then $[\![\mathcal{A}]\!] = \mathbb{L}$. Note that, we omit the transition label a, register guard* True *and the empty register update.*

Example 3.10. *Let $\mathbb{D} = \langle \mathsf{D}, \mathcal{R} \rangle$ be a data structure with a size function* $\mathrm{size} : \mathsf{D} \to \mathbb{N}$. *Let Σ be an alphabet and $\mathcal{A} = (\mathcal{L}, \mathsf{Reg}, \mathcal{L}_i, T, \mathcal{L}_f)$ an* RA *over Σ and $\mathbb{D}$. Let $\tilde{r} \in \mathsf{Reg}$ be a designated register. Assume that for every data word $w \in \mathbb{D}\Sigma^+$ we want to observe the maximal size of data which will be loaded into $\tilde{r}$ along any run with label w. For this, we consider the data semiring $\langle \mathrm{Arc}, \mathcal{F}_1 \rangle$ of Example 3.8 (2) and construct the* wRA $\mathcal{A}'$ *over Σ, $\mathbb{D}$ and $\mathbb{S}$ as follows. We take two copies $\mathcal{A}^{(1)}$ and $\mathcal{A}^{(2)}$ of $\mathcal{A}$ with the same set* Reg *of registers. The initial locations of $\mathcal{A}'$ are the initial locations of $\mathcal{A}^{(1)}$ whereas the final locations of $\mathcal{A}'$ are the final locations of $\mathcal{A}^{(2)}$. For all transitions t in these two copies we let $\mathrm{wt}_{\mathrm{trans}}(t) = 0$ and, for all registers $r \in \mathsf{Reg}$, we let $\mathrm{wt}_{\mathsf{data}}(t, r) = 0^{\mathsf{D}\times\mathsf{D}}$. Whenever there is a transition $\ell^{(1)} \xrightarrow{\varphi,\, a,\, \mathsf{up}} \ell^{(1)}$ in $\mathcal{A}^{(1)}$ with $\tilde{r} \in \mathsf{up}$, then we add to $\mathcal{A}'$ the transition $t = \big(\ell^{(1)} \xrightarrow{\varphi,\, a,\, \mathsf{up}} \ell^{(2)}\big)$ where $\ell^{(2)}$ is the location in $\mathcal{A}^{(2)}$ which corresponds to $\ell^{(1)}$. Let $f \in \mathcal{F}_1$ be defined as $f(d, d') = d'$. Then, we let $\mathrm{wt}_{\mathrm{trans}}(t) = 0$, $\mathrm{wt}_{\mathsf{data}}(t, \tilde{r}) = f$ and $\mathrm{wt}_{\mathsf{data}}(t, r) = 0^{\mathsf{D}\times\mathsf{D}}$ for all $r \in \mathsf{Reg} \setminus \{\tilde{r}\}$. Then, for all $w \in \mathbb{D}\Sigma^+$, $[\![\mathcal{A}']\!](w)$ is the maximal size of data which is loaded*

into $\tilde{r}$ along any run of $\mathcal{A}$ with label w. Note that if $\tilde{r}$ is not updated along any run with label w, then $[\![\mathcal{A}']\!](w) = -\infty$.

3.4 Relation to Weighted Timed Automata

In Section 3.2, we investigated the relation between register automata and timed automata in the qualitative setting. A semiring-based model for *weighted timed automata* (WTA) of [5, 13] was investigated in [85, 86]. In this section, we show that WTA over the timed semiring $\mathbb{S}$ can be simulated by WRA.

A *timed semiring* is a pair $\mathbb{S} = \langle \mathcal{S}, \mathcal{F} \rangle$ where $\mathcal{S} = (S, +, \cdot, \mathbf{0}, \mathbf{1})$ is a semiring and $\mathcal{F} \subseteq S^{\mathbb{R}_{\geq 0}}$ such that $\mathbf{1}^{\mathbb{R}_{\geq 0}} \in \mathcal{F}$. In the rest of this subsection, we assume that $\mathcal{S}$ is commutative.

Definition 3.11. *Let Σ be an alphabet. A* weighted timed automaton *(*WTA*) over Σ and $\mathbb{S}$ is a tuple $\mathcal{A} = (\mathcal{L}, C, \mathcal{L}_i, T, \mathcal{L}_f, \mathrm{wt})$ where $(\mathcal{L}, C, \mathcal{L}_i, T, \mathcal{L}_f)$ is a timed automaton over Σ (cf. Subsect. 3.2) and $\mathrm{wt} = \langle \mathrm{wt}_{\mathrm{trans}}, \mathrm{wt}_{\mathrm{time}} \rangle$ is a pair of weight functions such that $\mathrm{wt}_{\mathrm{trans}} : T \to S$ and $\mathrm{wt}_{\mathrm{time}} : \mathcal{L} \to \mathcal{F}$.*

Remark 3.12. *Note that, in contrast to* WRA*, time-dependent costs in* WTA *are assigned to locations and these costs are modeled by functions of a single argument.*

Let $\rho = \big(q_0 \vdash_{t_1,d_1} q_1 \vdash_{t_2,d_2} \ldots \vdash_{t_n,d_n} q_n\big)$ be a run of $\mathcal{A}$ where $q_i = (\ell_i, \nu_i)$ for all $i \in \{0, \ldots, n\}$. The *weight* of ρ is defined as

$$\mathrm{wt}(\rho) = \prod_{i=1}^{n} \mathrm{wt}_{\mathrm{time}}(\ell_{i-1})(d_i) \cdot \mathrm{wt}_{\mathrm{trans}}(t_i).$$

Note that $\mathrm{wt}(\rho) \in S$. The *behavior* of $\mathcal{A}$ is the mapping $[\![\mathcal{A}]\!] : \mathbb{T}\Sigma^+ \to S$ defined for all $w \in T\Sigma^+$ by

$$[\![\mathcal{A}]\!](w) = \sum \big(\mathrm{wt}(\rho) \mid \rho \text{ is a run of } \mathcal{A} \text{ with } \mathrm{label}(\rho) = w \big).$$

A mapping $\mathbb{L} : \mathbb{T}\Sigma^+ \to S$ is called a *timed series*. For $f : \mathbb{R}_{\geq 0} \to S$, let $\Psi(f) : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \to S$ be defined by

$$\Psi(f)(d, d') = \begin{cases} f(d' - d), & \text{if } d \leq d', \\ \mathbf{1}, & \text{otherwise.} \end{cases}.$$

To show that wTA over the timed semiring $\mathbb{S}$ can be simulated by wRA, we consider the data structure $\mathbb{D}^{\mathrm{Timed}}$ of Section. 3.2 and the data semiring $\tilde{\mathbb{S}} = \langle \mathcal{S}, \tilde{\mathcal{F}} \rangle$ with $\tilde{\mathcal{F}} = \{\Psi(f) \mid f \in \mathcal{F}\}$. Given a data series $\mathbb{L} : \mathbb{D}^{\mathrm{Timed}}\Sigma^+ \to S$ with $\mathbb{L}(w) = \mathbf{0}$ for all $w \in \mathbb{D}^{\mathrm{Timed}}\Sigma^+ \setminus \mathbb{T}\Sigma^+$, we identify $\mathbb{L}$ with the timed series $\tilde{L} = L|_{\mathbb{T}\Sigma^+}$.

Lemma 3.13. *Let $\mathbb{L} : \mathbb{T}\Sigma^+ \to S$ be a timed series recognizable by a* wTA *over Σ and $\mathbb{S}$. Then, $\mathbb{L}$ is recognizable by a* wRA *over Σ, $\mathbb{D}^{\mathrm{Timed}}$ and $\tilde{\mathbb{S}}$.*

Proof. Let $\mathcal{A}^{\mathrm{time}} = (\mathcal{L}, C, \mathcal{L}_i, T, \mathcal{L}_f, \langle \mathrm{wt}_{\mathrm{trans}}, \mathrm{wt}_{\mathrm{time}} \rangle)$ be a wTA over Σ and $\mathbb{S}$ such that $[\![\mathcal{A}^{\mathrm{time}}]\!] = \mathbb{L}$. We may assume that $T \neq \emptyset$. We define the weighted register automaton $\mathcal{A}^{\mathsf{data}} = (\mathcal{L}, \mathsf{Reg}, \mathcal{L}_i, T', \mathcal{L}_f, \langle \mathrm{wt}'_{\mathrm{trans}}, \mathrm{wt}'_{\mathsf{data}} \rangle)$ over Σ, $\mathbb{D}^{\mathrm{Timed}}$ and $\tilde{\mathbb{S}}$ as follows:

- $\mathsf{Reg} = C \cup \{\tilde{r}\}$ where $\tilde{r} \notin C$,
- T' consists of all transitions $t' = \big(\ell \xrightarrow{\tilde{\varphi} \wedge R_{\geq 0}\tilde{r},\, a,\, \mathsf{up} \cup \{\tilde{r}\}} \ell'\big)$ such that there exists a transition $t = \big(\ell \xrightarrow{\varphi,\, a,\, \Lambda} \ell'\big) \in T$ where $\Lambda = \mathsf{up}$ and $\tilde{\varphi} \in \Phi(\mathsf{Reg}, \mathbb{D}^{\mathrm{Timed}})$ is obtained from φ by replacing every constraint $x \bowtie k$ by $(R_{\bowtie k})x$. Then, we let:
  - $\mathrm{wt}'_{\mathrm{trans}}(t') = \mathrm{wt}_{\mathrm{trans}}(t)$
  - $\mathrm{wt}'_{\mathsf{data}}(t', r) = \mathbf{1}^{\mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0}}$ for all $r \in C$;
  - $\mathrm{wt}'_{\mathsf{data}}(t', \tilde{r}) = \Psi(\mathrm{wt}_{\mathrm{time}}(\ell))$.

It is easy to see that $[\![\mathcal{A}^{\mathsf{data}}]\!] = [\![\mathcal{A}^{\mathrm{time}}]\!]$. Hence the claim follows. □

Lemma 3.13 shows the extension of Lemma 3.5 to the weighted setting. However, in general, the statement analogous to Lemma 3.6 does not hold in the weighted setting.

Lemma 3.14. *There exist an alphabet Σ, a commutative timed semiring $\mathbb{S}$ with domain S and a timed series $\mathbb{L} : \mathbb{T}\Sigma^+ \to S$ such that $\mathbb{L}$ is recognizable by a* wRA *over Σ, $\mathbb{D}^{\mathrm{Timed}}$ and $\tilde{\mathbb{S}}$, but $\mathbb{L}$ is not recognizable by a timed automaton over Σ and $\mathbb{S}$.*

Proof. Let $\Sigma = \{a\}$ be a singleton alphabet and $\mathbb{S} = \langle \mathcal{S}, \mathcal{F} \rangle$ where $\mathcal{S}$ is the arctic semiring of Example 3.8(1) and $\mathcal{F} = \{0^{\mathbb{R}_{\geq 0}}, \exp\}$ with $\exp(t) = e^t$ for all $t \in \mathbb{R}_{\geq 0}$. For all $n \geq 1$, let $w_n = (a,1)(a,2)...(a,n)$. Then, we define $\mathbb{L}$ for all $w \in \mathbb{T}\Sigma^+$ as

$$\mathbb{L}(w) = \begin{cases} e^n, & \text{if } w = w_n \text{ for some } n \geq 1, \\ -\infty, & \text{otherwise} \end{cases}$$

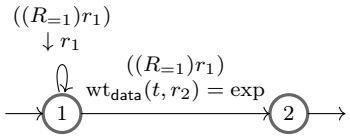


Figure 3.3: The wRA $\mathcal{A}^{\mathsf{data}}$ in the proof of Lemma 3.14

Consider the wRA $\mathcal{A}^{\mathsf{data}}$ over Σ, $\mathbb{D}^{\mathrm{Timed}}$ and $\tilde{\mathbb{S}}$ with two registers r_1, r_2 depicted in Fig. 3.3. In this figure, for all transitions t we have $\mathrm{wt}_{\mathrm{trans}}(t) = 0$, and for all registers $r \in \{r_1, r_2\}$, $\mathrm{wt}_{\mathsf{data}}(t, r) = 0^{\mathbb{R}_{\geq 0}}$ is not specified. Note that the register r_2 is never updated and its value is always 0. Then $[\![\mathcal{A}]\!]^{\mathsf{data}} = \mathbb{L}$.
Suppose now that there exists a wTA $\mathcal{A}^{\mathrm{time}}$ over Σ and $\mathbb{S}$ such that $[\![\mathcal{A}^{\mathrm{time}}]\!] = \mathbb{L}$. Then, there exists a constant $M \geq 1$ such that, for all $n \geq 1$ and every run ρ of $\mathcal{A}^{\mathrm{time}}$ with label w_n, we have:

$$\mathrm{wt}(\rho) \leq \sum_{i=1}^{n} (M + e) = (M + e)n$$

and hence $e^n = [\![\mathcal{A}^{\mathrm{time}}]\!](w_n) \leq (M + e)n$ for all $n \geq 1$. A contradiction. Then $\mathbb{L}$ is not recognizable by a wTA.

□

3.5 Closure Properties of Weighted Register Automata

In this section, we establish some basic closure properties for the class of recognizable data series. We will apply them in the proof of our logic characterization result. We note that the class of timed series recognizable by weighted timed automata of [86] is not stable under the Hadamard product even in the case of commutative semirings (cf. Example 5 of [86]). Interestingly, since we assign a data-dependent weight to every register, the class of recognizable data languages is closed under the Hadamard product in the case of commutative semirings.

Let Σ be an alphabet, $\mathbb{D} = \langle \mathsf{D}, \mathcal{R} \rangle$ a data structure and $\mathbb{S} = \langle (S, +, \cdot, \mathbf{0}, \mathbf{1}), \mathcal{F} \rangle$ a commutative data semiring over $\mathbb{D}$. Let $\mathbb{L}_1, \mathbb{L}_2 \in \mathbb{S}\langle\langle \mathbb{D}\Sigma^+ \rangle\rangle$ be data series. The *sum* $\mathbb{L}_1 + \mathbb{L}_2 \in \mathbb{S}\langle\langle \mathbb{D}\Sigma^+ \rangle\rangle$ and the *Hadamard product* $\mathbb{L}_1 \odot \mathbb{L}_2 \in \mathbb{S}\langle\langle \mathbb{D}\Sigma^+ \rangle\rangle$ are defined by

$(\mathbb{L}_1 \oplus \mathbb{L}_2)(w) = \mathbb{L}_1(w) + \mathbb{L}_2(w)$ respectively $(\mathbb{L}_1 \odot \mathbb{L}_2)(w) = \mathbb{L}_1(w) \cdot \mathbb{L}_2(w)$ for all $w \in \mathbb{D}\Sigma^+$.

Lemma 3.15. *Let Σ, Γ be alphabets, $\mathbb{D}$ a data semiring and $\mathbb{S}$ a commutative data semiring over $\mathbb{D}$. If $\mathbb{L}_1, \mathbb{L}_2 \in \mathbb{S}\langle\langle \mathbb{D}\Sigma^+ \rangle\rangle$ are recognizable, then so are $\mathbb{L}_1 \oplus \mathbb{L}_2$ and $\mathbb{L}_1 \odot \mathbb{L}_2$.*

Proof. The claim for $\mathbb{L}_1 \oplus \mathbb{L}_2$ can be easily shown by applying the standard disjoint union construction for automata. For $\mathbb{L}_1 \odot \mathbb{L}_2$, we establish an extension of the standard product construction of automata. For $j \in \{1, 2\}$, let $\mathcal{A}^{(j)} = (\mathcal{L}^{(j)}, \mathsf{Reg}^{(j)}, \mathcal{L}_i^{(j)}, T^{(j)}, \mathcal{L}_f^{(j)}, \langle \mathrm{wt}_{\mathrm{trans}}^{(j)}, \mathrm{wt}_{\mathsf{data}}^{(j)} \rangle)$ be a wRA with $[\![\mathcal{A}^{(j)}]\!] = \mathbb{L}_j$, and let $|\mathsf{Reg}^{(j)}| = m_j$ for $m_j \geqslant 1$. Without loss of generality, we assume that $\mathcal{L}^{(1)} \cap \mathcal{L}^{(2)} = \emptyset$ and $\mathsf{Reg}^{(1)} \cap \mathsf{Reg}^{(2)} = \emptyset$. We construct the wRA $\mathcal{A} = (\mathcal{L}, \mathsf{Reg}, \mathcal{L}_i, T, \mathcal{L}_f, \langle \mathrm{wt}_{\mathrm{trans}}, \mathrm{wt}_{\mathsf{data}} \rangle)$ for $\mathbb{L}_1 \odot \mathbb{L}_2$ as follows. We let $\mathcal{L} = \mathcal{L}^{(1)} \times \mathcal{L}^{(2)}$, $\mathsf{Reg} = \mathsf{Reg}^{(1)} \cup \mathsf{Reg}^{(2)}$, $\mathcal{L}_i = \mathcal{L}_i^{(1)} \times \mathcal{L}_i^{(2)}$ and $\mathcal{L}_f = \mathcal{L}_f^{(1)} \times \mathcal{L}_f^{(2)}$. We define T as the set of all transitions $t = \big((\ell_1, \ell_1') \xrightarrow{\varphi \wedge \varphi',\, a,\, \mathsf{up} \cup \mathsf{up}'} (\ell_2, \ell_2')\big)$ where $t_1 = \big(\ell_1 \xrightarrow{\varphi,\, a,\, \mathsf{up}} \ell_2\big) \in T_1$ and $t_2 = \big(\ell_1' \xrightarrow{\varphi',\, a,\, \mathsf{up}'} \ell_2'\big) \in T_2$. For such a transition t and a register $r \in \mathsf{Reg}$, we let $\mathrm{wt}_{\mathrm{trans}}(t) = \mathrm{wt}_{\mathrm{trans}}^{(1)}(t_1) \cdot \mathrm{wt}_{\mathrm{trans}}^{(2)}(t_2)$ and $\mathrm{wt}_{\mathsf{data}}(t, r) = \mathrm{wt}_{\mathsf{data}}^{(j)}(t_j, r)$ whenever $r \in \mathsf{Reg}^{(j)}$ for $j \in \{1, 2\}$. We want to show that $[\![\mathcal{A}]\!] = [\![\mathcal{A}^{(1)}]\!] \odot [\![\mathcal{A}^{(2)}]\!]$. First we show that there is a weight-preserving bijection between the set of runs of $\mathcal{A}$ and the set of pairs of runs of $\mathcal{A}_1$ and $\mathcal{A}_2$. Let $w \in \mathbb{D}\Sigma^+$ with $|w| = n$, for $n \geqslant 1$. Let

$$\rho ::= \; q_0 \vdash_{t_1, d_1} q_1 \vdash_{t_2, d_2} \ldots \vdash_{t_n, d_n} q_n$$

be a successful run of $\mathcal{A}$ on w such that, for $1 \leqslant i \leqslant n$, we have

$$t_i = \big((\ell_{i-1}, \ell_{i-1}') \xrightarrow{\varphi_i \wedge \varphi_i',\, a,\, \mathsf{up}_i \cup \mathsf{up}_i'} (\ell_i, \ell_i')\big) \in T.$$

With the given construction for $\mathcal{A}$ it can be seen that there are runs

$$\rho' ::= \; q_0' \vdash_{t_1', d_1} q_1' \vdash_{t_2', d_2} \ldots \vdash_{t_n', d_n} q_n'$$

of $\mathcal{A}_1$ and

$$\rho'' ::= \; q_0'' \vdash_{t_1'', d_1} q_1'' \vdash_{t_2'', d_2} \ldots \vdash_{t_n'', d_n} q_n''$$

of $\mathcal{A}_2$ over w, such that, for $1 \leqslant i \leqslant n$, we have

$$t_i' = \big(\ell_{i-1} \xrightarrow{\varphi_i,\, a,\, \mathsf{up}_i} \ell_i\big) \in T_1, \;\; t_i'' = \big(\ell_{i-1}' \xrightarrow{\varphi_i',\, a,\, \mathsf{up}_i'} \ell_i'\big) \in T_2.$$

Now using the definition for the weight functions and commutativity of the semiring, for a data word $w \in \mathbb{D}\Sigma^+$ we have:

$$\begin{aligned}
\mathrm{wt}(\rho) &= \prod_{i=1}^{n} \mathrm{wt}(q_{i-1} \vdash_{t_i,d_i} q_i) \\
&= \prod_{i=1}^{n} \prod_{j=1}^{m_1+m_2} \mathrm{wt}_{\mathsf{data}}(t_i, r_j)(\nu_i(r_j), d_i) \cdot \mathrm{wt}_{\mathrm{trans}}(t_i) \\
&= \prod_{i=1}^{n} \Big(\prod_{j=1}^{m_1} \mathrm{wt}_{\mathsf{data}}(t'_i, r_j)(\nu_i(r_j), d_i) \cdot \mathrm{wt}_{\mathrm{trans}}(t'_i) \\
&\qquad \cdot \prod_{j=1}^{m_2} \mathrm{wt}_{\mathsf{data}}(t''_i, r_j)(\nu_i(r_j), d_i) \cdot \mathrm{wt}_{\mathrm{trans}}(t'') \Big) \\
&= \prod_{i=1}^{n} \prod_{j=1}^{m_1} \mathrm{wt}_{\mathsf{data}}(t'_i, r_j)(\nu_i(r_j), d_i) \cdot \mathrm{wt}_{\mathrm{trans}}(t'_i) \\
&\cdot \prod_{i=1}^{n} \prod_{j=1}^{m_2} \mathrm{wt}_{\mathsf{data}}(t''_i, r_j)(\nu_i(r_j), d_i) \cdot \mathrm{wt}_{\mathrm{trans}}(t'') \\
&= \prod_{i=1}^{n} \mathrm{wt}(q'_{i-1} \vdash_{t'_i,d_i} q'_i) \cdot \prod_{i=1}^{n} \mathrm{wt}(q''_{i-1} \vdash_{t''_i,d_i} q''_i) = \mathrm{wt}(\rho_1) \cdot \mathrm{wt}(\rho_2).
\end{aligned}$$

Conversely, let ρ' and ρ'' as above be runs of $\mathcal{A}_1$ and $\mathcal{A}_2$, respectively, on w. One can see that the composition of ρ' and ρ'' gives the run ρ of $\mathcal{A}$ on w, and therefore with the same argument as above we have $\mathrm{wt}(\rho') \cdot \mathrm{wt}(\rho'') = \mathrm{wt}(\rho)$. We apply these arguments and hence for each $w \in \mathbb{D}\Sigma^+$ we have:

$$\begin{aligned}
[\![\mathcal{A}]\!](w) &= \sum \big(\mathrm{wt}(\rho) \mid \rho \in \mathrm{Run}_{\mathcal{A}}(w) \big) \\
&= \sum \big(\mathrm{wt}(\rho') \cdot \mathrm{wt}(\rho'') \mid \rho' \in \mathrm{Run}_{\mathcal{A}_1}(w) \text{ and } \rho'' \in \mathrm{Run}_{\mathcal{A}_2}(w) \big) \\
&= \sum \big(\mathrm{wt}(\rho') \mid \rho' \in \mathrm{Run}_{\mathcal{A}_1}(w) \big) \cdot \sum \big(\mathrm{wt}(\rho'') \mid \rho'' \in \mathrm{Run}_{\mathcal{A}_2}(w) \big) \\
&= ([\![\mathcal{A}^{(1)}]\!] \odot [\![\mathcal{A}^{(2)}]\!])(w) \\
&= (\mathbb{L}_1 \odot \mathbb{L}_2)(w).
\end{aligned}$$

Note that the third equality is implied by the arguments above and also by using distributivity and commutativity of the data semiring.

□

Let Γ, Σ be alphabets and $\pi : \Gamma \to \Sigma$ a mapping. For a data word $u = (a_1, d_1)...(a_n, d_n) \in \mathbb{D}\Gamma^+$, let $\pi(u) = (\pi(a_1), d_1)...(\pi(a_n), d_n) \in \mathbb{D}\Sigma^+$.

For a data series $\mathbb{L} \in \mathbb{S}\langle\langle \mathbb{D}\Gamma^+ \rangle\rangle$, the *projection* $\pi(\mathbb{L}) \in \mathbb{S}\langle\langle \mathbb{D}\Sigma^+ \rangle\rangle$ is defined for all $w \in \mathbb{D}\Sigma^+$ by $\pi(\mathbb{L})(w) = \sum \big(\mathbb{L}(u) \mid u \in \mathbb{D}\Gamma^+ \text{ and } \pi(u) = w\big)$. For a data series $\mathbb{L} \in \mathbb{S}\langle\langle \mathbb{D}\Sigma^+ \rangle\rangle$, the *inverse projection* $\pi^{-1}(\mathbb{L}) \in \mathbb{S}\langle\langle \mathbb{D}\Gamma^+ \rangle\rangle$ is defined for all $u \in \mathbb{D}\Gamma^+$ by $\pi^{-1}(\mathbb{L})(u) = \mathbb{L}(\pi(u))$.

Lemma 3.16. *Let Σ, Γ be alphabets, $\mathbb{D}$ a data semiring and $\mathbb{S}$ a data semiring over $\mathbb{D}$, and $\pi : \Gamma \to \Sigma$ a mapping.*

1. *If $\mathbb{L} \in \mathbb{S}\langle\langle \mathbb{D}\Gamma^+ \rangle\rangle$ is recognizable, then so is $\pi(\mathbb{L})$.*
2. *If $\mathbb{L} \in \mathbb{S}\langle\langle \mathbb{D}\Sigma^+ \rangle\rangle$ is recognizable, then so is $h^{-1}(\mathbb{L})$.*

Proof. 1. The construction of a wRA $\mathcal{A}$ for $\pi(\mathbb{L})$ is based on the same idea as in Lemma 1 of [47]. Let $\mathcal{A}' = (\mathcal{L}', \mathsf{Reg}', \mathcal{L}'_i, T', \mathcal{L}'_f, \langle \mathrm{wt}'_{\mathrm{trans}}, \mathrm{wt}'_{\mathsf{data}} \rangle)$ be a wRA over the alphabet Γ with $[\![\mathcal{A}']\!] = \mathbb{L}$. We construct a wRA $\mathcal{A} = (\mathcal{L}, \mathsf{Reg}, \mathcal{L}_i, T, \mathcal{L}_f, \langle \mathrm{wt}_{\mathrm{trans}}, \mathrm{wt}_{\mathsf{data}} \rangle)$ over Σ for $\pi(\mathbb{L})$ as follows: We let $\mathcal{L} = \mathcal{L}' \times \Gamma$, $\mathsf{Reg} = \mathsf{Reg}'$, $\mathcal{L}_i = \mathcal{L}'_i \times \{a_0\}$ for a fixed $a_0 \in \Gamma$ and $\mathcal{L}_f = \mathcal{L}'_f \times \Gamma$. We also let T consist of all transitions $t = ((\ell, \gamma') \xrightarrow{\varphi,\, \pi(\gamma),\, \mathsf{up}} (\ell', \gamma))$ where $t' = (\ell \xrightarrow{\varphi,\, \gamma,\, \mathsf{up}} \ell')$ is a transition in T'. For such a transition t and a register $r \in \mathsf{Reg}$ we let $\mathrm{wt}_{\mathrm{trans}}(t) = \mathrm{wt}'_{\mathrm{trans}}(t')$ and $\mathrm{wt}_{\mathsf{data}}(t, r) = \mathrm{wt}'_{\mathsf{data}}(t', r)$ for the corresponding transition $t' \in T'$. Now we show that $[\![\mathcal{A}]\!] = \pi([\![\mathcal{A}']\!])$. For a data word $u = (\gamma_1, d_1)...(\gamma_n, d_n) \in \mathbb{D}\Gamma^+$, let $w = (\pi(\gamma_1), d_1)...(\pi(\gamma_n), d_n) \in \mathbb{D}\Sigma^+$. Consider the run $\rho = q_0 \vdash_{t_1,d_1} q_1 \vdash_{t_2,d_2} ... \vdash_{t_n,d_n} q_n$ of $\mathcal{A}$ on w such that $t_i = ((\ell_{i-1}, \gamma_{i-1}) \xrightarrow{\varphi_i,\, \pi(\gamma_i),\, \mathsf{up}_i} (\ell_i, \gamma_i))$ for $i \in \{1, 2, ..., n\}$. By the given construction for $\mathcal{A}$, clearly $\rho' = q_0 \vdash_{t'_1,d_1} q_1 \vdash_{t'_2,d_2} ... \vdash_{t'_n,d_n} q_n$, where $t'_i = (\ell_{i-1} \xrightarrow{\varphi_i,\, \gamma_i,\, \mathsf{up}_i} \ell_i)$ is a run of $\mathcal{A}'$ on $u \in h^{-1}(w)$. Now, we can define the mapping

$$\Omega : \mathrm{Run}_{\mathcal{A}}(w) \to \bigcup_{u \in \pi^{-1}(w)} \mathrm{Run}_{\mathcal{A}'}(u)$$

by $\Omega(\rho) = \rho'$. It is easy to see that Ω is a bijection. In addition, by the definition of the weight functions we have $\mathrm{wt}(\rho') = \mathrm{wt}(\Omega(\rho))$. Therefore,

$$\begin{aligned}
[\![\mathcal{A}]\!](w) &= \sum \big(\, \mathrm{wt}(\rho) \mid \rho \in \mathrm{Run}_{\mathcal{A}}(w)\big) \\
&= \sum \big(\, \mathrm{wt}(\Omega^{-1}(\rho')) \mid \rho' \in \bigcup_{u \in \pi^{-1}(w)} \mathrm{Run}_{\mathcal{A}'}(u)\big) \\
&= \sum_{u \in \pi^{-1}(w)} \sum \big(\, \mathrm{wt}(\rho') \mid \rho' \in \mathrm{Run}_{\mathcal{A}'}(u)\big) \\
&= \sum_{u \in \pi^{-1}(w)} [\![\mathcal{A}']\!](u) = \pi([\![\mathcal{A}']\!])(w) = \pi(\mathbb{L})(w)
\end{aligned}$$

which shows that $[\![\mathcal{A}]\!] = \pi(\mathbb{L})$ and hence $\pi(\mathbb{L})$ is recognizable.

2. This case is straightforward: the location space is preserved and every transition of a wRA for $\mathbb{L}$ with label $a \in \Sigma$ is simulated in a wRA for $\pi^{-1}(\mathbb{L})$ by several transitions with labels from $\pi^{-1}(a)$. □

3.6 Weighted Existential MSO-Logic

In this section we introduce weighted existential monadic second-order (wEMSO) logic over data semirings for data words augmented with binary data functions. Then we show that a suitable fragment of our weighted logic and our weighted register automata model are expressively equivalent. Our new logic will be based on the ideas of Wilke and Bouyer [103, 22] of logical characterizations of timed and data automata and on the approach of Droste and Gastin [41] to weighted logic over semirings.

As in [19], in order to describe easily boolean properties, we introduce two levels of formulas: boolean and weighted. We operate with the boolean formulas as in the usual logic. On the weighted level, we add weights and binary functions from a data semiring and extend the logical operations by computations in the data semiring.

Let $\mathcal{V}_1$ and $\mathcal{V}_2$ be finite pairwise disjoint sets of first-order and second-order variables. Let $\mathcal{V} = \mathcal{V}_1 \cup \mathcal{V}_2$. Let Σ be an alphabet, $\mathbb{D} = (\mathsf{D}, \mathcal{R})$ a data structure with the initial data value $\perp \in \mathsf{D}$ and $\mathbb{S} = \langle (S, +, \cdot, \mathbf{0}, \mathbf{1}), \mathcal{F} \rangle$ a data semiring over $\mathbb{D}$. *Weighted first-order logic* $\mathrm{wFO}[\Sigma, \mathbb{D}, \mathbb{S}]$ over Σ, $\mathbb{D}$ and $\mathbb{S}$ is defined by the grammar

$$\begin{aligned} \beta &::= P_a(x) \mid x \le y \mid x \in X \mid R(X, x) \mid \beta \vee \beta \mid \neg\beta \mid \exists x.\beta \\ \varphi &::= \beta \mid s \mid f(x, y) \mid f(\perp, y) \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x.\varphi \mid \forall x.\varphi \end{aligned}$$

where $a \in \Sigma$, $x, y \in \mathcal{V}_1$, $X \in \mathcal{V}_2$, $R \in \mathcal{R}$, $s \in S$ and $f \in \mathcal{F}$. The formulas β are called *boolean* over Σ and $\mathbb{D}$. Let $\mathrm{Bool}[\Sigma, \mathbb{D}]$ denote the set of all boolean formulas. We note that a formula $R(X, x)$ relates to a relative distance formula of Wilke [103] and reflects the performance of register r: here d is an initial value of the register r and the second-order variable X keeps track of positions where r is updated; moreover, at the position x the register guard Rr is checked. Using boolean formulas, we define the boolean formulas $x < y$, $x = y$, $x \notin X$, $\beta_1 \wedge \beta_2$, $\forall x.\beta$, $\beta_1 \to \beta_2$ and $\beta_1 \leftrightarrow \beta_2$ as usual.

Weighted existential MSO logic wEMSO$[\Sigma, \mathbb{D}, \mathbb{S}]$ over Σ, $\mathbb{D}$ and $\mathbb{S}$ is defined to be the set of all formulas of the form $\exists X_1 ... \exists X_n . \varphi$ where $n \geq 0$, $X_1, ..., X_n$ are second-order variables and $\varphi \in$ wFO$[\Sigma, \mathbb{D}, \mathbb{S}]$. Given a formula $\psi \in$ wEMSO$[\Sigma, \mathbb{D}, \mathbb{S}]$, the set *free*$(\psi) \subseteq \mathcal{V}$ of *free variables* of ψ is defined as usual. We say that ψ is a *sentence* if *free*$(\psi) = \emptyset$.

Let $w = (a_1, d_1)...(a_n, d_n) \in \mathbb{D}\Sigma^+$ be a data word. Let $\text{dom}(w) = \{1, ..., n\}$, the *domain* of w. A $(\mathcal{V}, w)$-*assignment* is a mapping

$$\sigma : \mathcal{V} \to \text{dom}(w) \cup 2^{\text{dom}(w)}$$

mapping first-order variables to elements in dom(w) and second-order variables to subsets of dom(w). For a first-order variable x and a position $i \in \text{dom}(w)$, the $(\mathcal{V}, w)$-assignment $\sigma[x/i]$ is defined on $\mathcal{V} \setminus \{x\}$ as σ, and we let $\sigma[x/i](x) = i$. We also let $\sigma[x/i] \restriction_{\mathcal{V}\setminus\{x\}} = \sigma \restriction_{\mathcal{V}\setminus\{x\}}$. For a second-order variable X and $I \subseteq \text{dom}(w)$, the $(\mathcal{V}, w)$-assignment $\sigma[X/I]$ is defined similarly. Given a formula $\beta \in \text{Bool}[\Sigma, \mathbb{D}]$ and a $(\mathcal{V}, w)$-assignment σ, the satisfaction relation $(w, \sigma) \models \beta$ is defined by induction on the structure of β as usual where, for formulas of the form $R(X, x)$, we let $(w, \sigma) \models R(X, x)$ iff, letting $d_0 = \perp$, for the greatest position $i \in \sigma(X) \cup \{0\}$ with $i < \sigma(x)$ we have $(d_i, d_{\sigma(x)}) \in R$. Since the satisfaction relation depends only on values of free variables, we will abuse notation and also write $(w, \sigma|_U) \models \beta$ for any $U \subseteq \mathcal{V}$ with *free*$(\beta) \subseteq U$.

Let $\mathbb{D}\Sigma^+_{\mathcal{V}}$ denote the set of all pairs (w, σ) where $w \in \mathbb{D}\Sigma^+$ and σ is a $(\mathcal{V}, w)$-assignment. Given a formula $\psi \in$ wEMSO$[\Sigma, \mathbb{D}, \mathbb{S}]$, the *semantics* of ψ is the mapping $[\![\psi]\!]_{\mathcal{V}} : \mathbb{D}\Sigma^+_{\mathcal{V}} \to S$ defined for all $(w, \sigma) \in \mathbb{D}\Sigma^+_{\mathcal{V}}$ with $w = (a_1, d_1)...(a_n, d_n)$ as shown in Table 3.1. If ψ is a sentence, then we can ignore the $(\mathcal{V}, w)$-assignments in the definition of the semantics and consider it as the data series $[\![\psi]\!] : \mathbb{D}\Sigma^+ \to S$.

The following lemma shows that for each formula $\psi \in$ wEMSO$[\Sigma, \mathbb{D}, \mathbb{S}]$, the semantics for the different finite sets $\mathcal{V}$ of variables containing *free*(ψ) are consistent with each other. Since the proof is similar to the existing result for different weighted structures (see [41]), we do not give the proof here.

Lemma 3.17. *Let $\psi \in$ wEMSO$[\Sigma, \mathbb{D}, \mathbb{S}]$ and $\mathcal{V}$ a finite set of variables with free$(\psi) \subseteq \mathcal{V}$. Then $[\![\psi]\!]_{\mathcal{V}}(w, \sigma) = [\![\psi]\!](w, \sigma \restriction_{free(\psi)})$ for any valid $(w, \sigma) \in \mathbb{D}\Sigma^+_{\mathcal{V}}$. In addition, $[\![\psi]\!]$ is recognizable if and only if $[\![\psi]\!]_{\mathcal{V}}$ is recognizable.*

$$
\begin{array}{lcl}
[\![\beta]\!](w,\sigma) & = & \begin{cases} \mathbf{1}, & \text{if } (w,\sigma) \models \beta, \\ \mathbf{0}, & \text{otherwise} \end{cases} \\
[\![s]\!](w,\sigma) & = & s \\
[\![f(x,y)]\!](w,\sigma) & = & f(d_{\sigma(x)}, d_{\sigma(y)}) \\
[\![f(\bot,y)]\!](w,\sigma) & = & f(\bot, d_{\sigma(y)}) \\
[\![\varphi_1 \vee \varphi_2]\!](w,\sigma) & = & [\![\varphi_1]\!](w,\sigma) + [\![\varphi_2]\!](w,\sigma) \\
[\![\varphi_1 \wedge \varphi_2]\!](w,\sigma) & = & [\![\varphi_1]\!](w,\sigma) \cdot [\![\varphi_2]\!](w,\sigma) \\
[\![\exists x.\varphi]\!](w,\sigma) & = & \sum_{i \in \mathrm{dom}(w)} [\![\varphi]\!](w, \sigma[x/i]) \\
[\![\forall x.\varphi]\!](w,\sigma) & = & \prod_{i \in \mathrm{dom}(w)} [\![\varphi]\!](w, \sigma[x/i]) \\
[\![\exists X.\varphi]\!](w,\sigma) & = & \sum_{I \subseteq \mathrm{dom}(w)} [\![\varphi]\!](w, \sigma[X/I])
\end{array}
$$

Table 3.1: The semantics of wEMSO-formulas

Example 3.18. *Consider the data series $\mathbb{L}$ of Example 3.9. Note that $\mathbb{L}$ is definable by the* wEMSO$[\Sigma, \mathbb{D}, \mathbb{S}]$*-sentence*

$$\exists X.\exists x.[(X = \{x\}) \wedge \forall (y > x).([R(X,y) \wedge 1] \vee \neg R(X,y))]$$

where $X = \{x\}$ is an abbreviation for the formula $\forall z.(z \in X \leftrightarrow z = x)$, $\forall (y > x).\varphi$ abbreviates the formula $\forall y.((y > x \wedge \varphi) \vee (y \leq x))$, and $R = (=_D)$.

3.7 Restricted wEMSO-Logic

Our goal is to study the connection between wRA and our new weighted logic on data words. Similarly to the result of [41], the unrestricted use of formulas of the form $\forall x.\varphi$ leads to unrecognizable data series. Below we give a further example of unrecognizability which is specific for wEMSO on data words.

Example 3.19. *Let $\Sigma = \{a\}$ be a singleton alphabet and $\mathbb{D}$ be a data structure with the data domain $\mathbb{N}$. Consider the data semiring $\mathbb{S} = \langle \mathrm{Arc}, \mathcal{F} \rangle$ where* Arc *is the arctic semiring of Example 3.8 (1) and $\mathcal{F} = \{0^{\mathbb{N}\times\mathbb{N}}, f\}$ where $f : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ is defined by $f(n,n') = n$ for all $n, n' \in \mathbb{N}$. Consider the sentence $\varphi \in$ wEMSO$[\Sigma, \mathbb{D}, \mathbb{S}]$ defined by $\varphi = \exists x.\exists y.f(x,y)$. Note that, for every $n \in \mathbb{N}$ and the data word $w_n = (a,n) \in \mathbb{D}\Sigma^+$ of length 1, we have $[\![\varphi]\!](w_n) = n$. Then for any* wRA *$\mathcal{A}$ over Σ, $\mathbb{D}$ and $\mathbb{S}$, there exists a constant $M \in \mathbb{N}$ such that $[\![\mathcal{A}]\!](w_n) \leq M$ for all $n \in \mathbb{N}$. Hence, $[\![\mathcal{A}]\!] \neq [\![\varphi]\!]$.*

Now we will investigate a fragment of wEMSO which is expressively equivalent to wRA. Example 3.19 shows that the use of binary functions

$f(x,y)$ is not suitable for a logical characterization by wRA. We follow the approach of Wilke [103] for a logical characterization of timed automata where the use of every expressive time distance predicate $\mathrm{dist}(x,y) \bowtie k$ with $x,y \in \mathcal{V}_1$, $\bowtie \in \{<,\leq,=,\geq,>\}$ was restricted to relative time distance predicate $\mathrm{dist}(X,y) \bowtie k$ with $X \in \mathcal{V}_2$ (note that the relative time distance predicates correspond to the formulas $R(X,y)$ in our logic $\mathrm{Bool}[\Sigma,\mathbb{D}]$). We replace the formulas $f(x,y)$ by the formulas $f(X,y)$ whose semantics is defined in a similar manner as for $R(X,y)$, as follows: For $f \in \mathcal{F}$, a first-order variable x and a second-order variable X, let $f(X,x)$ denote the $\mathrm{wFO}[\Sigma,\mathbb{S},\mathbb{D}]$-formula $\exists y.(\beta(X,x,y) \wedge f(y,x)) \vee (\beta'(X,x) \wedge f(\bot,x))$ where $\beta(X,x,y)$ is the boolean formula $y \in X \wedge y < x \wedge \forall z.([y < z \wedge z < x] \rightarrow z \notin X)$ and $\beta'(X,x)$ is the boolean formula $\forall y.(y < x \rightarrow y \notin X)$. Then, for all $(w,\sigma) \in \mathbb{D}\Sigma^+$ with $w = (a_1,d_1)...(a_n,d_n)$, we have $[\![f(X,x)]\!](w,\sigma) = f(d_i,d_{\sigma(x)})$ where $i \in \sigma(X) \cup \{0\}$ is the greatest position with $i < \sigma(x)$ and $d_0 = \bot$.

The following example shows that the use of our new logical operator $f(X,x)$ in the scope of a weighted quantifier $\forall y$ with $y \neq x$ also goes beyond recognizability by wRA.

Example 3.20. *Let Σ and $\mathbb{D}$ be defined as in Example 3.19. Consider the data semiring $\mathbb{S} = \langle \mathrm{Arc}, \mathcal{F} \rangle$ with $\mathcal{F} = \{0^{\mathbb{N}\times\mathbb{N}}, f\}$ where f is defined for all $n,n' \in \mathbb{N}$ by $f(n,n') = n'$. Consider the sentence $\varphi \in \mathrm{wEMSO}[\Sigma,\mathbb{D},\mathbb{S}]$ defined by $\varphi = \exists X.\exists x.(\beta(X,x) \wedge \forall y.f(X,x))$ where $y \neq x$ and the boolean formula $\beta(X,x)$ is defined as $\forall y.y \leq x \wedge \forall y.(y \in X \leftrightarrow \forall z.y \leq z)$. Note that $\beta(X,x)$ describes that x is the last position of a data word and X is the singleton set containing the first position of a data word. For any $n \geq 2$ and the data word $w_n = (a,0)^{n-1}(a,n) \in \mathbb{D}\Sigma^+$, we have $[\![\varphi]\!](w_n) = n^2$. Then for any wRA $\mathcal{A}$ over Σ, $\mathbb{D}$ and $\mathcal{S}$, there exists a constant $M \in \mathbb{N}$ such that $[\![\mathcal{A}]\!](w_n) \leq M \cdot n$ for all $n \geq 2$. Hence, $[\![\mathcal{A}]\!] \neq [\![\varphi]\!]$.*

Now, based on the explanations above, we will define the desired fragment of wEMSO for wRA. Note that, similarly to [41], we must restrict the use of $\forall x$ to simplified formulas without weighted quantifiers. Let x be a first-order variable. We say that a formula $\gamma \in \mathrm{wFO}[\Sigma,\mathbb{D},\mathbb{S}]$ is *almost boolean over x* if it is derived by the grammar

$$\gamma \ ::= \ \beta \mid s \mid f(X,x) \mid \gamma \vee \gamma \mid \gamma \wedge \gamma$$

where $\beta \in \mathrm{Bool}[\Sigma,\mathbb{D}]$, $s \in S$, $f \in \mathcal{F}$ and X is a second-order variable. Let $\mathrm{aBool}(x)[\Sigma,\mathbb{D},\mathbb{S}]$ denote the set of all almost boolean formulas over x. Then, *restricted weighted first-order logic* $\mathrm{wFO}^{\mathrm{res}}[\Sigma,\mathbb{D},\mathbb{S}] \subseteq \mathrm{wFO}[\Sigma,\mathbb{D},\mathbb{S}]$

is defined by the grammar

$$\varphi \ ::= \ \beta \mid s \mid f(X,x) \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists x.\varphi \mid \forall x.\gamma$$

where $\beta \in \mathrm{Bool}[\Sigma, \mathbb{D}]$, $s \in S$, $f \in \mathcal{F}$, x is a first-order variable, X is a second-order variable and $\gamma \in \mathrm{aBool}(x)[\Sigma, \mathbb{D}, \mathbb{S}]$. *Restricted weighted existential MSO logic* $\mathrm{wEMSO}^{\mathrm{res}}[\Sigma, \mathbb{D}, \mathbb{S}] \subseteq \mathrm{wEMSO}[\Sigma, \mathbb{D}, \mathbb{S}]$ is defined to be the set of all formulas of the form $\exists X_1 ... \exists X_n.\varphi$ where $n \geq 0$, $X_1, ..., X_n$ are second-order variables and $\varphi \in \mathrm{wFO}^{\mathrm{res}}[\Sigma, \mathbb{D}, \mathbb{S}]$. We say that a fragment $\mathrm{Frag} \subseteq \mathrm{wEMSO}[\Sigma, \mathbb{D}, \mathbb{S}]$ is *expressively equivalent to* wRA if, for every data series $\mathbb{L} : \mathbb{D}\Sigma^+ \to S$, $\mathbb{L}$ is recognizable by a wRA over Σ, $\mathbb{D}$ and $\mathbb{S}$ iff $\mathbb{L}$ is definable by a sentence in Frag.

Theorem 3.21. *Let Σ be an alphabet, $\mathbb{D}$ a data structure, and $\mathbb{S}$ a commutative data semiring over $\mathbb{D}$. Then* $\mathrm{wEMSO}^{\mathrm{res}}[\Sigma, \mathbb{D}, \mathbb{S}]$ *is expressively equivalent to* wRA.

Remark 3.22. *Note that a logical characterization of weighted timed automata was given in [86], Theorems 30 and 39. The structure of the syntactically restricted logical fragment of our Theorem 3.21 has the following advantages. First, we do not restrict the use of weighted conjunctions $\varphi_1 \wedge \varphi_2$ and we only need to restrict the use of $\forall$-quantifiers to formulas without weighted quantifiers. Second, in the syntactically restricted weighted logic of [86] over non-idempotent semirings (cf. Theorem 39 of [86]), the use of $\forall$-quantifier is restricted to formulas which define timed languages of* bounded variability*, a notion introduced by Wilke [104]. Intuitively, the variability of a timed word corresponds to the maximum number of events that may occur within one time unit. Since we deal with register automata, the same notion cannot be applied here, and therefore we need a new technique. We will prove Theorem 3.21 in Section 3.9.*

3.8 Visibly Register Automata

Two of the main difficulties of the proof of Theorem 3.21 are that register automata are neither determinizable nor closed under complement. The goal of this section is to investigate a subclass of register automata which can be applied in the proof of our logical characterization result. Our determinizable subclass could be also of independent interest.

The idea is to make the register updates visible in transition labels. Note that a similar idea was applied in *event-clock automata* [3] and *visibly pushdown automata* [4]. Since the class of languages recognizable by

event-clock automata is not closed under projections of input symbols [3], this model is not suitable for the translation of logical formulas. We use a slightly different approach. Recall that in event-clock automata, the input alphabet is arbitrary and with every letter a clock is associated. In contrast, in our model we take an arbitrary set of registers and an input alphabet is defined depending on this set of registers.

Throughout all of this section, we fix an alphabet Σ, a data structure $\mathbb{D} = (\mathsf{D}, \mathcal{R})$ and a finite set of *registers* Reg. Let $\Sigma^{\langle \mathsf{Reg} \rangle}$ denote the alphabet $\Sigma \times \{0,1\}^{\mathsf{Reg}}$.

Definition 3.23. *A* visibly register automaton *over* Σ, $\mathbb{D}$ *and* Reg *is a register automaton* $\mathcal{A}$ *over* $\Sigma^{\langle \mathsf{Reg} \rangle}$ *and* $\mathbb{D}$ *with the set of registers* Reg *such that, for every transition* $q \xrightarrow{\varphi,\,(a,\theta),\,\mathsf{up}} q'$ *of* $\mathcal{A}$ *where* q, q' *are locations,* $a \in \Sigma$, $\theta \in \{0,1\}^{\mathsf{Reg}}$, $\mathsf{up} \subseteq \mathsf{Reg}$ *and* φ *is a register guard, we have* $\mathsf{up} = \{r \in \mathsf{Reg} \mid \theta(r) = 1\}$.

Note that $\mathcal{A}$ recognizes the language $L(\mathcal{A}) \subseteq \mathbb{D}\big(\Sigma^{\langle \mathsf{Reg} \rangle}\big)^+$, visibly register automata form a subclass of register automata.

We say that a register automaton $\mathcal{A}$ over Σ and $\mathbb{D}$ is *deterministic* if it has a single initial location and whenever $\ell \xrightarrow{\varphi,\,a,\,\mathsf{up}} \ell'$ and $\ell \xrightarrow{\varphi',\,a',\,\mathsf{up}'} \ell''$ are two distinct transitions of $\mathcal{A}$, then φ and φ' are mutually exclusive, i.e., for all registers valuations ν and all data values $d \in \mathsf{D}$, we have $(\nu, d) \not\models \varphi \wedge \varphi'$. We call $\mathcal{A}$ *complete* if for all locations ℓ of $\mathcal{A}$, all letters $a \in \Sigma$, all register valuations ν and all data values $d \in \mathsf{D}$, there exists a transition $\ell \xrightarrow{\varphi,\,a,\,\mathsf{up}} \ell'$ of $\mathcal{A}$ with $(\nu, d) \models \varphi$.

The following lemma shows that the class of visibly register automata is determinizable.

Theorem 3.24. *Let* $\mathcal{A}$ *a visibly register automaton over* Σ, $\mathbb{D}$ *and* Reg. *Then, there exists a deterministic and complete visibly register automaton* $\mathcal{A}'$ *over* Σ, $\mathbb{D}$ *and* Reg *such that* $L(\mathcal{A}) = L(\mathcal{A}')$.

Proof. The proof follows a similar idea as the proof of Theorem 1 of [3] about determinization of event-clock automata. Let $\mathcal{A} = (\mathcal{L}, \mathsf{Reg}, \mathcal{L}_i, T, \mathcal{L}_f)$ be a visibly register automaton over the alphabet Σ recognizing the data language $L(\mathcal{A})$. From $\mathcal{A}$ we construct a visibly register automaton $\mathcal{A}' = (\mathcal{L}', \mathsf{Reg}, \mathcal{L}'_i, T', \mathcal{L}'_f)$ with $L(\mathcal{A}') = L(\mathcal{A})$ as follows:

- $\mathcal{L}' = \mathcal{P}(\mathcal{L})$, $\mathcal{L}'_i = \{\mathcal{L}_i\}$, $\mathcal{L}'_f = \{U \subseteq \mathcal{L} \mid U \cap \mathcal{L}_f \neq \emptyset\}$.

- T' is defined as follows. Suppose that $U \in \mathcal{P}(\mathcal{L})$ and $(a,\theta) \in \Sigma^{\langle \mathsf{Reg} \rangle}$. Let $(t_i)_{i\in\{1,...,m\}}$ be an enumeration of the set of all transitions in T with label (a,θ) starting in a location from U. For each $i \in \{1,...,m\}$ let $t_i = \big(\ell_i \xrightarrow{\varphi_i,\,(a,\theta),\,\mathsf{up}} \ell_i'\big)$ with the update set $\mathsf{up} = \{r \in \mathsf{Reg} \mid \theta(r) = 1\}$. For any subset $J \subseteq \{1,...,m\}$, we add to T' the transition $t' = \big(U \xrightarrow{\varphi_J,\,(a,\theta),\,\mathsf{up}} U'\big)$ where $U' = \{\ell_i' \mid i \in J\}$ with $\varphi_J = \bigwedge_{i\in J} \varphi_i \wedge \bigwedge_{i\notin J} \neg\varphi_i$.

We show that

1. $\mathcal{A}'$ is deterministic and complete;
2. $L(\mathcal{A}') = L(\mathcal{A})$.

First we show (1), The completeness of the automaton $\mathcal{A}'$ is implied by the given construction. Now we show that $\mathcal{A}'$ is deterministic. Assume that $t_1' = \big(U \xrightarrow{\varphi_{J_1},\,(a,\theta),\,\mathsf{up}} U_1'\big) \in T'$ and $t_2' = \big(U \xrightarrow{\varphi_{J_2},\,(a,\theta),\,\mathsf{up}} U_2'\big) \in T'$ with $t_1' \neq t_2'$. Then, for the subsets $J_1, J_2 \subseteq \{1,2,...,m\}$ we have $J_1 \neq J_2$ and hence φ_{J_1} and φ_{J_2} are mutually exclusive. Therefore $\mathcal{A}'$ is deterministic.

Now we show (2). Let $u = ((a_1,\theta_1),d_1)...((a_n,\theta_n),d_n) \in L(\mathcal{A})$. Then there exists a run

$$\rho = \langle \ell_0,\nu_0\rangle \vdash_{t_1,d_1} \langle \ell_1,\nu_1\rangle \vdash_{t_2,d_2} ... \vdash_{t_n,d_n} \langle \ell_n,\nu_n\rangle \tag{3.1}$$

of $\mathcal{A}$ on u where $t_i = \big(\ell_{i-1} \xrightarrow{\varphi_i,\,(a_i,\theta_i),\,\mathsf{up}_i} \ell_i\big)$ with $\mathsf{up}_i = \{r \in \mathsf{Reg} \mid \theta_i(r) = 1\}$ for all $i \in \{1,...,n\}$. Note that $\langle p_0,\nu_0\rangle$ is the initial state and all the register valuations $\nu_1,...,\nu_n$ are uniquely determined by ν_0 and u. Now consider the run

$$\rho' = \langle U_0,\nu_0\rangle \vdash_{t_1',d_1} \langle U_1,\nu_1\rangle \vdash_{t_2',d_2} ... \vdash_{t_n',d_n} \langle U_n,\nu_n\rangle \tag{3.2}$$

where $t_i' = \big(U_{i-1} \xrightarrow{\varphi_{J_i},\,(a_i,\theta_i),\,\mathsf{up}_i} U_i\big)$ with $\mathsf{up}_i = \{r \in \mathsf{Reg} \mid \theta_i(r) = 1\}$ for all $i \in \{1,...,n\}$. Note that $U_0 = \mathcal{L}_i' = \{\mathcal{L}_i\}$ and each U_i is obtained based on the idea in the construction explained above. By the definition of the transitions t_i', $i \in \{1,...,n\}$, we can see that $\ell_i \in U_i$ and therefore $t_i' \in T'$. In addition, since $\ell_n \in \mathcal{L}_f$ we have $\mathcal{L}_f \cap U_n \neq \emptyset$ and so $U_n \in \mathcal{L}_f'$. Thus, ρ' is a run of $\mathcal{A}'$ on u. Hence, $u \in L(\mathcal{A}')$. It implies that $L(\mathcal{A}) \subseteq L(\mathcal{A}')$.

Conversely, let $u = ((a_1,\theta_1),d_1)...((a_n,\theta_n),d_n) \in L(\mathcal{A}')$. Then there exists a successful run of the form 3.2 on w where $t_i' = \big(U_{i-1} \xrightarrow{\varphi_{J_i},\,(a_i,\theta_i),\,\mathsf{up}_i} U_i\big)$

with $\mathsf{up}_i = \{r \in \mathsf{Reg} \mid \theta_i(r) = 1\}$ for all $i \in \{1, ..., n\}$. Then, there exist $\ell_0 \in \mathcal{L}_i$, $\ell_1, ..., \ell_{n-1} \in \mathcal{L}$ and $\ell_n \in \mathcal{L}_f$ such that $t_i = \left(\ell_{i-1} \xrightarrow{\varphi_i,\, (a_i,\theta_i),\, \mathsf{up}_i} \ell_i\right) \in T$, for $i \in \{1, ..., n\}$. Thus, we can define a run of the form 3.1 of $\mathcal{A}$ on u. Hence $u \in L(\mathcal{A})$. Therefore, $L(\mathcal{A}') \subseteq L(\mathcal{A})$. Hence (2) holds true. □

Using Theorem 3.24 for the complement, it is not difficult to verify the closure properties for visibly register automata stated in the next lemma.

Lemma 3.25. *The class of data languages recognizable by visibly register automata over Σ, $\mathbb{D}$ and Reg is closed under union, intersection and complement.*

Proof. For union, we apply the standard automata-theoretic disjoint union construction. Now we give the proof for the complement. By Theorem 3.24, there exists a deterministic and complete visibly register automaton $\mathcal{A} = (\mathcal{L}, \mathsf{Reg}, \mathcal{L}_i, T, \mathcal{L}_f)$ over Σ such that $L(\mathcal{A}) = L \subseteq \mathbb{D}(\Sigma^{\langle \mathsf{Reg} \rangle})^+$. Then, we can define the visibly register automaton $\mathcal{A}' = (\mathcal{L}, \mathsf{Reg}, \mathcal{L}_i, T, \mathcal{L} \backslash \mathcal{L}_f)$. Since $\mathcal{A}$ is deterministic, for all $u \in L$, there exists only one successful run ρ of $\mathcal{A}$ on u which ends in a location in $\mathcal{L}_f$. It means, $u \notin L(\mathcal{A}')$. In addition, since $\mathcal{A}$ is complete, for all $u \in \mathbb{D}\Sigma^+ \backslash L$ there exists a run which ends in a location from $\mathcal{L} \backslash \mathcal{L}_f$. Hence, $u \in L(\mathcal{A}')$. It implies that $L(\mathcal{A}') = \mathbb{D}(\Sigma^{\langle \mathsf{Reg} \rangle})^+ \backslash L$. Finally, the claim for the intersection is implied by the closure under complement and union. □

Let Γ, Δ be alphabets and $\pi : \Gamma \to \Delta$ a mapping. We extend π to the projection $\pi : \mathbb{D}(\Gamma^{\langle \mathsf{Reg} \rangle})^+ \to \mathbb{D}(\Delta^{\langle \mathsf{Reg} \rangle})^+$ by applying π only to the Σ-components of data words, i.e., for a data word $u = ((a_1, \theta_1), d_1)...((a_n, \theta_n), d_n) \in \mathbb{D}(\Gamma^{\langle \mathsf{Reg} \rangle})^+$ we can define the data word $\pi(u) = ((\pi(a_1), \theta_1), d_1)...((\pi(a_n), \theta_n), d_n) \in \mathbb{D}(\Delta^{\langle \mathsf{Reg} \rangle})^+$. Then for a data language $L \subseteq \mathbb{D}(\Gamma^{\langle \mathsf{Reg} \rangle})^+$, the projection $\pi(L)$ is defined by $\{\pi(u) \mid u \in L\} \subseteq \mathbb{D}(\Delta^{\langle \mathsf{Reg} \rangle})^+$. Similarly, for a data language $L' \subseteq \mathbb{D}(\Delta^{\langle \mathsf{Reg} \rangle})^+$, we define the inverse projection $\pi^{-1}(L')$ by $\{u \in \mathbb{D}(\Gamma^{\langle \mathsf{Reg} \rangle})^+ \mid \pi(u) \in L'\} \subseteq \mathbb{D}(\Gamma^{\langle \mathsf{Reg} \rangle})^+$.

Lemma 3.26. *Let Γ, Δ be alphabets and $\pi : \Gamma \to \Delta$ a mapping.*

1. *If $L \subseteq \mathbb{D}(\Gamma^{\langle \mathsf{Reg} \rangle})^+$ is recognizable by a visibly register automaton over Γ, $\mathbb{D}$ and Reg, then $\pi(L)$ is recognizable by a visibly register automaton over Δ, $\mathbb{D}$ and Reg.*

2. *If $L \subseteq \mathbb{D}(\Delta^{\langle \mathsf{Reg} \rangle})^+$ is recognizable by a visibly register automaton over Δ, $\mathbb{D}$ and Reg, then $\pi^{-1}(L)$ is recognizable by a visibly register automaton over Γ, $\mathbb{D}$ and Reg.*

Proof. Let Γ and Δ be alphabets, and $\pi : \Gamma \to \Delta$ a mapping.

1. Let $\mathcal{A} = (\mathcal{L}, \mathsf{Reg}, \mathcal{L}_i, T, \mathcal{L}_f)$ be a visibly register automaton over Γ, $\mathbb{D}$ and Reg recognizing the data language $L \subseteq \mathbb{D}(\Gamma^{\langle \mathsf{Reg} \rangle})^+$. We construct a visibly register automaton $\mathcal{A}' = (\mathcal{L}, \mathsf{Reg}, \mathcal{L}_i, T', \mathcal{L}_f)$ with the same location space $\mathcal{L}$ over Δ, $\mathbb{D}$ and Reg recognizing $\pi(L)$ as follows: the set of transitions T' consists of all transitions $t' = \big(\ell \xrightarrow{\varphi,\ (\pi(a),\theta),\ \mathsf{up}} \ell'\big)$ where $t = \big(\ell \xrightarrow{\varphi,\ (a,\theta),\ \mathsf{up}} \ell'\big)$ is a transition in T. Now one can see that $L(\mathcal{A}') = \pi(L)$.

2. Let $\mathcal{B} = (\mathcal{L}, \mathsf{Reg}, \mathcal{L}_i, T, \mathcal{L}_f)$ be a visibly register automaton over Δ, $\mathbb{D}$ and Reg recognizing the data language $L \subseteq \mathbb{D}(\Delta^{\langle \mathsf{Reg} \rangle})^+$. We obtain a visibly register automaton $\mathcal{B}' = (\mathcal{L}, \mathsf{Reg}, \mathcal{L}_i, T', \mathcal{L}_f)$ over Γ, $\mathbb{D}$ and Reg recognizing $\pi^{-1}(L) \subseteq \mathbb{D}(\Gamma^{\langle \mathsf{Reg} \rangle})^+$ by defining T' as follows: the set of transitions T' consists of all transitions $t' = \big(\ell \xrightarrow{\varphi,\ (a,\theta),\ \mathsf{up}} \ell'\big)$ where $t = \big(\ell \xrightarrow{\varphi,\ (\pi(a),\theta),\ \mathsf{up}} \ell'\big)$ is a transition in T. It can be seen that $L(\mathcal{B}') = \pi^{-1}(L)$. □

Now let $\beta \in \mathrm{Bool}[\Sigma, \mathbb{D}]$ be a formula and Reg the set of all second-order variables X occurring in a subformula of β of the form $R(X, x)$. Using the standard encoding of free variables, we encode the set of all pairs $(w, \sigma|_{\mathit{free}(\beta)})$ such that $(w, \sigma) \in \mathbb{D}\Sigma^+$ and $(w, \sigma) \models \beta$ as the data language $L(\varphi) \subseteq \mathbb{D}(\Gamma^{\langle \mathsf{Reg} \rangle})^+$ where $\Gamma = \Sigma \times \{0,1\}^{\mathit{free}(\beta) \setminus \mathsf{Reg}}$. Using Lemmas 3.25 and 3.26 and Theorem 3.24, we will show by induction the following theorem.

Theorem 3.27. *Let $\beta \in \mathrm{Bool}[\Sigma, \mathbb{D}]$ be a formula and Reg the set of all second-order variables X occurring in a subformula of β of the form $R(X, x)$. Then, there exists a deterministic visibly register automaton $\mathcal{A}$ over Γ, $\mathbb{D}$ and Reg such that $L(\mathcal{A}) = L(\beta)$.*

Proof. Let $\beta \in \mathrm{Bool}[\Sigma, \mathbb{D}]$ and Reg the set of all second-order variables X occurring in a subformula of β of the form $R(X, x)$. We show that for each formula β, the language $L(\beta)$ can be recognized by a deterministic visibly register automaton $\mathcal{A}$ over the extended alphabet $\Gamma^{\langle \mathsf{Reg} \rangle}$ where $\Gamma = \Sigma \times \{0,1\}^{\mathit{free}(\beta) \setminus \mathsf{Reg}}$, $\mathbb{D}$ and Reg.

1. If β is one of the formulas $P_a(x)$, $x \leqslant y$ or $x \in X$, we can construct a deterministic visibly register automaton $\mathcal{A}_\beta$ using the same approach as for the formulas in MSO.

2. The automaton $\mathcal{A}_{R(X,x)}$ for the formula $\beta = R(X, x)$ over the alphabet $\Gamma^{\langle X \rangle}$ where $\Gamma = \Sigma \times \{0,1\}^{\{x\}}$ is depicted in Fig. 3.4. Here, Σ_{ij} with

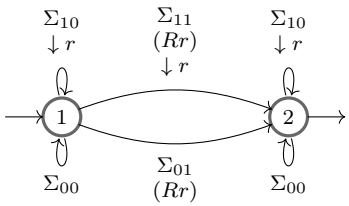


Figure 3.4: The visibly register automaton $\mathcal{A}$ for the formula $R(X, x)$

$i, j \in \{0, 1\}$ means any letter in $\Gamma^{\langle X \rangle}$ whose X-component is i and x-component is j and $r = X$ is the register of $\mathcal{A}_{R(X,x)}$. This automaton can guess at which step the last transition labeled with a letter in $\Gamma^{\langle X \rangle}$ with a 1 in the X-row is taken and updates the register r at this transition. Then it checks that whenever a transition is labeled with a letter including a 1 in the x-row, then the data value loaded into the register at the last position labeled with a letter including a 1 in the X-row satisfies the guard Rr.

3. If β is one of the formulas $\beta \vee \beta$, $\neg\beta$ or $\exists x.\beta$, we use the standard approach as for the formulas in MSO, applying Lemma 3.25 and 3.26.

□

3.9 Definability Equals Recognizability

In this section, we give the proof of Theorem 3.21. First, we show that recognizability implies definability.

Theorem 3.28. *Let $\mathcal{A}$ be a* wRA *over Σ, $\mathbb{D}$ and $\mathbb{S}$. Then there exists a sentence $\varphi \in \mathrm{wEMSO}^{\mathrm{res}}[\Sigma, \mathbb{D}, \mathbb{S}]$ such that $[\![\varphi]\!] = [\![\mathcal{A}]\!]$.*

Proof. Let $\mathcal{A} = (\mathcal{L}, \mathsf{Reg}, \mathcal{L}_i, T, \mathcal{L}_f, \langle \mathrm{wt}_{\mathrm{trans}}, \mathrm{wt}_{\mathsf{data}} \rangle)$. First, using $\mathrm{Bool}[\Sigma, \mathbb{D}]$-formulas, we describe runs of $\mathcal{A}$. For this, we fix an enumeration $(t_i)_{1 \le i \le n}$ of T and an enumeration $(r_j)_{1 \le j \le m}$ of Reg. Assume that $t_i = \big(\ell_i \xrightarrow{\varphi_i,\, a_i,\, \mathsf{up}_i} \ell_i'\big)$, for $1 \le i \le n$. Then, we associate with every transition t_i a second-order variable X_i which keeps track of positions where t_i is taken and associate with every register r_j a second-order variable Y_j which keeps track of positions where r_j is updated. Now, a run of $\mathcal{A}$ can be described using the formula $\beta \in \mathrm{Bool}[\Sigma, \mathbb{D}]$ with the set of free variables $\mathit{free}(\beta) = \{X_1, ..., X_n, Y_1, ..., Y_m\}$ defined by

$$\beta = \mathrm{Partition} \wedge \mathrm{Labels} \wedge \mathrm{Initial} \wedge \mathrm{Consistent} \wedge \mathrm{Final} \wedge \mathrm{Registers}$$

where

• Partition $= \forall x. \bigvee_{1 \leq i \leq n}((x \in X_i) \wedge \bigwedge_{j \neq i}(x \notin X_j))$ demands that values of variables $X_1, ..., X_n$ form a parition of the domain of an input data word.
• Labels $= \forall x. \bigwedge_{a \in \Sigma}(P_a(x) \rightarrow \bigvee(x \in X_i \mid 1 \leq i \leq n,\ a_i = a))$ demands that the transition labels of a run are compatible with an input data word.
• Initial $= \exists x.(\forall y.(x \leq y) \wedge \bigvee(x \in X_i \mid 1 \leq i \leq n,\ q_i \in \mathcal{L}_i))$ demands that a run starts in an initial location from $\mathcal{L}_i$.
• Consistent $= \forall x. \forall y.((y = x + 1) \rightarrow \bigvee(x \in X_i \wedge y \in X_j \mid 1 \leq i, j \leq n, q_i' = q_j))$ demands that every two successive transitions are matching, i.e., they are connected via the same location.
• Final $= \exists x.(\forall y.(y \leq x) \wedge \bigvee(x \in X_i \mid 1 \leq i \leq n,\ q_i \in \mathcal{L}_f))$ demands that a run ends in a final location from $\mathcal{L}_f$.
• For all $x \in \mathcal{V}_1$ and $\varphi \in \Phi(\mathsf{Reg}, \mathbb{D})$, let $f_x(\varphi) \in \mathrm{Bool}[\Sigma]$ be obtained from φ by replacing every subformula Rr_j with $R \in \mathcal{R}$ and $j \in \{1, ..., m\}$ by $R(Y_j, x)$. Then Registers $= \forall x. \bigvee_{1 \leq i \leq n}(x \in X_i \wedge \bigwedge_{j \in \mathsf{up}_i} x \in Y_j \wedge \bigwedge_{j \in \mathsf{Reg} \setminus \mathsf{up}_i} x \notin Y_j \wedge f_x(\varphi_i))$ checks that whenever a transition t_i is taken (i.e. $x \in X_i$), then each register $j \in \mathsf{up}_i$ is updated (i.e., $x \in Y_j$), and each register $j \notin \mathsf{up}_i$ is not updated (i.e., $x \notin Y_j$), and a register valuation and a new data value must satisfy the register guard φ_i (i.e., $f_x(\varphi_i)$ holds true).

Then, the behavior of $\mathcal{A}$ can be described using the $\mathrm{wEMSO}^{\mathrm{res}}[\Sigma, \mathbb{D}, \mathbb{S}]$-sentence $\varphi = \exists X_1, ..., X_n, Y_1, ..., Y_m.(\beta \wedge \forall x.[\bigvee_{i=1}^{n}(x \in X_i \wedge \mathrm{wt}_{\mathrm{trans}}(t_i) \wedge \bigwedge_{j=1}^{m} \mathrm{wt}_{\mathsf{data}}(t_i, r_j)(Y_j, x))])$, i.e., $[\![\varphi]\!] = [\![\mathcal{A}]\!]$. □

Now we show that definability by sentences in $\mathrm{wEMSO}^{\mathrm{res}}[\Sigma, \mathbb{D}, \mathbb{S}]$ implies recognizability by wRA. Our proof will follow a similar strategy as the proof of the corresponding theorem in [41], i.e., we proceed by induction on the structure of the formula, encode the values of variables as letters of an extended alphabet and apply closure properties stated in Lemma 3.15. A crucial problem occurs with the $\forall x$-quantifiers and this case requires a new proof technique, since unweighted register automata are not determinizable and our almost boolean formulas contain functions of the form $f(X, x)$. We solve this problem by translating a $\mathrm{wEMSO}^{\mathrm{res}}$-sentence into a sentence where $\forall x$-quantifiers are applied to formulas of the simplified form. Then, using our Theorem 3.27, we can construct a wRA for $\forall x$-formulas.

Let Σ be an alphabet, $\mathbb{D}$ a data structure and $\mathbb{S} = \langle (S, +, \cdot, \mathbf{0}, \mathbf{1}), \mathcal{F} \rangle$ a data semiring over $\mathbb{D}$. Let $x \in \mathcal{V}_1$ be a first-order variable. We say that a formula $\kappa \in \mathrm{wFO}[\Sigma, \mathbb{S}, \mathbb{D}]$ is a *semi-granular weight formula* over $\Sigma, \mathbb{S}, \mathbb{D}$ and x if it is of the form $s \wedge f_1(X_1, x) \wedge ... \wedge f_r(X_r, x)$ where $s \in S$, $r \geq 0$, $f_1, ..., f_r \in \mathcal{F}$ and $X_1, ..., X_r$ are second-order variables. If $X_1, ..., X_r$

are pairwise distinct, then κ is called a *granular weight formula*. Let $\mathrm{Gran}(x)[\Sigma, \mathbb{S}, \mathbb{D}]$ denote the set of all granular weight formulas over $\Sigma, \mathbb{S}, \mathbb{D}$ and x. We say that a formula $\gamma \in \mathrm{aBool}(x)[\Sigma, \mathbb{S}, \mathbb{D}]$ is a *simple almost boolean formula* over $\Sigma, \mathbb{S}, \mathbb{D}$ and x if it is of the form $\bigvee_{i=1}^{n}(\beta_i \wedge \kappa_i)$ where $n \geq 1$, $\kappa_1, ..., \kappa_n \in \mathrm{Gran}(x)[\Sigma, \mathbb{S}, \mathbb{D}]$ and $\beta_1, ..., \beta_n \in \mathrm{Bool}[\Sigma, \mathbb{D}]$ are boolean formulas. We say that a formula $\psi \in \mathrm{wEMSO}^{\mathrm{res}}[\Sigma, \mathbb{S}, \mathbb{D}]$ is *canonical* over $\Sigma, \mathbb{S}, \mathbb{D}$ if whenever it contains a subformula of the form $\forall x.\gamma$, then γ is a simple almost boolean formula over $\Sigma, \mathbb{S}, \mathbb{D}$ and x. Now we show that each sentence $\psi \in \mathrm{wEMSO}^{\mathrm{res}}[\Sigma, \mathbb{S}, \mathbb{D}]$ can be translated into a canonical sentence over $\Sigma, \mathbb{S}, \mathbb{D}$:

Lemma 3.29. *Let $\psi \in \mathrm{wEMSO}^{\mathrm{res}}[\Sigma, \mathbb{S}, \mathbb{D}]$ be a sentence. Then, there exists a canonical sentence ζ over $\Sigma, \mathbb{S}, \mathbb{D}$ such that $[\![\zeta]\!] = [\![\psi]\!]$*

Proof. First, using the commutativity and distributivity of the data semiring $\mathbb{S}$, we claim that we can replace every almost boolean formula $\gamma \in \mathrm{aBool}(x)[\Sigma, \mathbb{S}, \mathbb{D}]$ occurring in ψ by a formula $\gamma' = \bigvee_{i=1}^{n}(\beta_i \wedge \kappa_i)$ where $\beta_1, ..., \beta_n \in \mathrm{Bool}[\Sigma, \mathbb{D}]$ and each κ_i is a semi-granular weight formula over $\Sigma, \mathbb{S}, \mathbb{D}$ and x which is of the form $s_i \wedge \bigwedge_{k=1}^{r} f_{ik}(Y_k, x)$. We prove this by induction on the structure of γ. If γ is a boolean formula $\beta \in \mathrm{Bool}[\Sigma, \mathbb{D}]$, it can be replaced by the formula $\beta \wedge \mathbf{1}$ which is a formula of the desired form. In case γ is the constant $s \in S$ we replace it by the formula $\mathsf{True} \wedge s$. In case $\gamma = f(Y, x)$, where $f \in \mathcal{F}$, Y is a second-order variable and x is a first-order variable, we replace it by the formula $\mathbf{1} \wedge \mathsf{True} \wedge f(Y, x)$ which is also of the desired form. Now we let $\gamma_1 = \bigvee_{i=1}^{m}(\beta_i \wedge \kappa_i)$ and $\gamma_2 = \bigvee_{i=m+1}^{n}(\beta_i \wedge \kappa_i)$ where $\beta_i, ..., \beta_n \in \mathrm{Bool}[\Sigma, \mathbb{D}]$ and each κ_i is a semi-granular weight formula over $\Sigma, \mathbb{S}, \mathbb{D}$ and x which is of the form $s_i \wedge \bigwedge_{k=1}^{r} f_{ik}(Y_k, x)$. Assume that $\gamma = \gamma_1 \vee \gamma_2$. Then clearly we have $\gamma_1 \vee \gamma_2 = \bigvee_{i=1}^{n}(\beta_i \wedge \kappa_i)$ which is a semi-granular weight formula over $\Sigma, \mathbb{S}, \mathbb{D}$ and x. Now let γ_1 be as above, and let $\gamma_2 = \bigvee_{j=1}^{n}(\beta'_j \wedge \kappa'_j)$ where $\beta'_1, ..., \beta'_n \in \mathrm{Bool}[\Sigma, \mathbb{D}]$ and each κ'_i is a semi-granular weight formula over $\Sigma, \mathbb{S}, \mathbb{D}$ and x which is of the form $s'_j \wedge \bigwedge_{k=1}^{r} f'_{jk}(Y_k, x)$. Assume that $\gamma = \gamma_1 \wedge \gamma_2$. Then we have $\gamma_1 \wedge \gamma_2 = \bigvee_{i=1}^{m}\bigvee_{j=1}^{n}(\beta_i \wedge \beta'_j) \wedge (\kappa_i \wedge \kappa'_j)$ where

$$\begin{aligned}\kappa_i \wedge \kappa'_j &= \big(s_i \wedge \bigwedge_{k=1}^{r} f_{ik}(Y_k, x)\big) \wedge \big(s'_j \wedge \bigwedge_{k=1}^{r} f'_{jk}(Y_k, x)\big) \\ &= (s_i \wedge s'_j) \wedge \bigwedge_{k=1}^{r} \big(f_{ik}(Y_k, x) \wedge f'_{jk}(Y_k, x)\big).\end{aligned}$$

Therefore, $\gamma_1 \wedge \gamma_2$ is also a semi-granular weight formula over $\Sigma, \mathbb{S}, \mathbb{D}$ and x. Note that the second equality is obtained by the commutativity of $\mathbb{S}$.

Now let $\eta \in \text{wEMSO}^{\text{res}}[\Sigma, \mathbb{S}, \mathbb{D}]$ be the sentence obtained after these replacements.

Second, we replace semi-granular weight formulas in η by granular weight formulas. The idea is the following. Assume that $\eta = \exists X_1, ..., X_k.\varphi$ with $\varphi \in \text{wFO}^{\text{res}}[\Sigma, \mathbb{S}, \mathbb{D}]$. In the case when φ contains a semi-granular formula $\kappa = s \wedge f_1(Y_1, x) \wedge ... \wedge f_l(Y_l, x) \wedge ... \wedge f_j(Y_j, x) \wedge ... \wedge f_r(Y_r, x)$ with $l \neq j$ and $Y_l = Y_j$, then we take a fresh second-order variable Z and replace η by the sentence $\exists X_1, ..., X_k, Z.([\forall z.(z \in Z \leftrightarrow z \in Y_l)] \wedge \tilde{\varphi})$ where $\tilde{\varphi}$ is obtained from φ by replacing the variable Y_j in κ by the fresh variable Z. Following this process, in finitely many steps we can get rid of all repeating second-order variables in semi-granular weight formulas. We call this formula ζ. Now for a data word $(w, \sigma) \in \mathbb{D}\Sigma^+$ we have:

$$\begin{aligned}
[\![\zeta]\!] &= [\![\exists X_1, ..., X_k, Z.([\forall z.(z \in Z \leftrightarrow z \in Y_l)] \wedge \tilde{\varphi})]\!](w, \sigma) \\
&= \sum_{I \subseteq \text{dom}(w)} [\![\exists X_1, ..., X_k([\forall z.(z \in Z \leftrightarrow z \in Y_l)] \wedge \tilde{\varphi})]\!](w, \sigma[Z/I]) \\
&= [\![\exists X_1, ..., X_k([\forall z.(z \in Z \leftrightarrow z \in Y_l)] \wedge \tilde{\varphi})]\!](w, \sigma[Z/\sigma(Y_l)]) \\
&= [\![\exists X_1, ..., X_k.\tilde{\varphi})]\!](w, \sigma[Z/\sigma[Y_l]]) \\
&= [\![\exists X_1, ..., X_k.\varphi]\!](w, \sigma) = [\![\psi]\!].
\end{aligned}$$

This completes the proof. □

Theorem 3.30. *Let $\psi \in \text{wEMSO}^{\text{res}}[\Sigma, \mathbb{S}, \mathbb{D}]$ be a sentence. Then there exists a* wRA *$\mathcal{A}$ over Σ, $\mathbb{S}$ and $\mathbb{D}$ such that $[\![\mathcal{A}]\!] = [\![\psi]\!]$.*

We will prove this theorem by induction on the structure of a subformula ζ of ψ. As usual, whenever ζ contains free variables, ζ will be translated into a wRA over the extended alphabet $\Sigma \times \{0, 1\}^{free(\zeta)}$. Now we show the following lemmas:

Lemma 3.31. *Let Σ be an alphabet, $\mathbb{D}$ a data structure and $\mathbb{S} = \langle (S, +, \cdot, \mathbf{0}, \mathbf{1}), \mathcal{F} \rangle$ a data semiring over $\mathbb{D}$. The semantics of all boolean formulas* Bool$[\Sigma, \mathbb{D}]$*, all formulas $s \in S$ and $f(X, x) \in \mathcal{F}$ are recognizable by a* wRA*.*

Proof. Let $\beta \in \text{Bool}[\Sigma, \mathbb{D}]$. By Theorem 3.27, we can construct a deterministic visibly register automaton $\mathcal{A}$ for β which will be considered as an RA. It can be easily transformed into a desired wRA for β by using $\mathbf{1}$ for wt_{trans} and $\mathbf{1}^{\mathsf{D} \times \mathsf{D}}$ for $\text{wt}_{\mathsf{data}}$. The automaton construction for the constant s is straightforward. For the formula $f(X, x)$ over the alphabet $\Gamma^{\langle X \rangle}$ where $\Gamma = \Sigma \times \{0, 1\}^{\{x\}}$, the automaton is depicted in Fig. 3.5. The idea behind

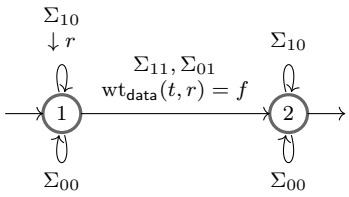


Figure 3.5: The WRA $\mathcal{A}$ for the formula $f(X, x)$

the construction is the same as the one for the formula $R(X, x)$. Note that in Fig. 3.5, for the transition from the location 1 to the location 2, $\mathrm{wt}_{\mathrm{trans}}$ is $\mathbf{1}$ and $\mathrm{wt}_{\mathsf{data}}$ is the weight function f. For all the other transitions we assign $\mathbf{1}$ to $\mathrm{wt}_{\mathrm{trans}}$ and $\mathbf{1}^{\mathsf{D}\times\mathsf{D}}$ to $\mathrm{wt}_{\mathsf{data}}$.

□

Lemma 3.32. *Let $\zeta_1, \zeta_2 \in \mathrm{wFO}[\Sigma, \mathbb{S}, \mathbb{D}]$ such that $[\![\zeta_1]\!]$ and $[\![\zeta_2]\!]$ are recognizable. Then $[\![\zeta_1 \vee \zeta_2]\!]$ and $[\![\zeta_1 \wedge \zeta_2]\!]$ are also recognizable.*

Proof. Let $\mathcal{V} = \mathit{free}(\zeta_1 \vee \zeta_2) = \mathit{free}(\zeta_1 \wedge \zeta_2) = \mathit{free}(\zeta_1) \cup \mathit{free}(\zeta_2)$. By applying Lemma 3.17, we can see that $[\![\zeta_1]\!]_{\mathcal{V}}$ and $[\![\zeta_2]\!]_{\mathcal{V}}$ are both recognizable. Now by Lemma 3.15 (1), $[\![\zeta_1 \vee \zeta_2]\!] = [\![\zeta_1]\!]_{\mathcal{V}} \vee [\![\zeta_2]\!]_{\mathcal{V}}$ and by Lemma 3.15 (2), $[\![\zeta_1 \wedge \zeta_2]\!] = [\![\zeta_1]\!]_{\mathcal{V}} \wedge [\![\zeta_2]\!]_{\mathcal{V}}$ are recognizable. □

Lemma 3.33. *Let $\zeta' \in \mathrm{wFO}[\Sigma, \mathbb{S}, \mathbb{D}]$ such that $[\![\zeta']\!]$ is recognizable. Then $[\![\exists x.\zeta']\!]$ and $[\![\exists X.\zeta']\!]$ are recognizable.*

Proof. Consider the formula $\zeta = \exists x.\zeta'$ for $\zeta' \in \mathrm{wEMSO}^{\mathrm{res}}[\Sigma, \mathbb{S}, \mathbb{D}]$ such that $[\![\zeta']\!]$ is recognizable. Let $\mathcal{V} = \mathit{free}(\exists x.\zeta')$. Note that $x \notin \mathcal{V}$. Consider the projection $h : \Sigma^+_{\mathcal{V}\cup\{x\}} \to \Sigma^+_{\mathcal{V}}$ which erases the x-row in $\Sigma_{\mathcal{V}\cup\{x\}}$. Then for a word $(w, \sigma) \in \mathbb{D}\Sigma^+_{\mathcal{V}\cup\{x\}}$ where σ is a valid $(\mathcal{V}, w)$-assignment, we have

$$\begin{aligned}
[\![\exists x.\zeta']\!](w, \sigma) &= \sum([\![\zeta']\!](w, \sigma[x/i]) \mid i \in \mathrm{dom}(w)) \\
&= \sum([\![\zeta']\!](w, \sigma') \mid \pi(w, \sigma') = (w, \sigma)) \\
&= \pi([\![\zeta']\!])(w, \sigma).
\end{aligned}$$

Due to Lemma 3.16 (1), $[\![\exists x.\zeta']\!]$ is recognizable. The proof for the case $\exists X.\zeta'$ is similar. □

Lemma 3.34. *Let $\gamma \in \mathrm{aBool}(x)[\Sigma, \mathbb{S}, \mathbb{D}]$ be a simple almost boolean formula over $\Sigma, \mathbb{S}, \mathbb{D}$ and x. Then $\forall x.\gamma$ is recognizable.*

Proof. Let $\zeta = \forall x.\gamma$. Let Reg be the set of all second-order variables Y such that γ has a subformula of the form $R(Y,x)$ with $R \in \mathcal{R}$ or $f(Y,x)$ with $f \in \mathcal{F}$. Let $(Y_k)_{1\leq k\leq r}$ be an enumeration of Reg. Now by Lemma 3.29, we can transform γ into the form

$$\gamma' = \bigvee_{i=1}^{n}(\beta_i \wedge s_i \wedge \bigwedge_{k=1}^{r} f_{ik}(Y_k,x))$$

where $\beta_1,...,\beta_n \in \mathrm{Bool}[\Sigma,\mathbb{D}]$, $s_i \in S$ and $f_{ik} \in \mathcal{F}$. We also may assume for simplicity that, for all $i \in \{1,...,n\}$ and $k \in \{1,...,r\}$, β_i has a subformula of the form $R(Y_k,x)$. This means that $\mathsf{Reg} \subseteq \mathit{free}(\beta_i)$ for all $i \in \{1,...,n\}$. Let $\tilde{S} \subseteq S$ be the set of all s_i appearing in γ' and $\tilde{\mathcal{F}} \subseteq \mathcal{F}$ the set of all f_{ik} appearing in γ'. Consider the extended alphabet $\Delta = \Sigma \times \{1,...,n\} \times \tilde{S} \times \tilde{\mathcal{F}}^r$. A data word in $\mathbb{D}\Delta^+_{\mathit{free}(\zeta)}$ is written by $(w,u,v,\vec{g},\sigma)$ where $(w,\sigma) \in \mathbb{D}\Sigma^+_{\mathit{free}(\zeta)}$, $u = u_1...u_{|w|} \in \{1,...,n\}^+$, $v = v_1...v_{|w|} \in \tilde{S}^+$ and $\vec{g} = (g^{(1)},...,g^{(n)}) \in (\tilde{\mathcal{F}}^+)^r$ and $g^{(i)} = g^{(i)}_1...g^{(i)}_{|w|} \in \tilde{\mathcal{F}}^+$ for all $i \in \{1,...,r\}$. We let $\xi \in \mathrm{Bool}[\Delta,\mathbb{D}]$ be of the form

$$\xi = \forall x.\textstyle\bigvee_{i=1}^{n}\Big(\bigvee_{a,v,g_1,...,g_r} P_{(a,i,v,g_1,...,g_r)}(x) \to \tilde{\beta}_i \wedge \bigvee_{a} P_{(a,i,s_i,f_{i1},...,f_{ir})}(x)\Big)$$

where each formula $\tilde{\beta}_i \in \mathrm{Bool}[\Delta,\mathbb{D}]$ is obtained from β_i by replacing every predicate $P_a(x)$ with $a \in \Sigma$ and $x \in \mathcal{V}_1$ by the formula

$$\bigvee \big(P_{(a,u,v,g_1,...,g_r)}(x) \mid u \in \{1,...,n\}, v \in \tilde{S}, g_1,...,g_r \in \tilde{\mathcal{F}}\big).$$

Note that $\mathit{free}(\xi) = \mathit{free}(\zeta)$ and, in particular, $\mathsf{Reg} \subseteq \mathit{free}(\xi)$. By Theorem 3.27, there exists a deterministic visibly register automaton $\mathcal{A} = (\mathcal{L}, \mathsf{Reg}, \mathcal{L}_i, T, F)$ over $\Delta \times \{0,1\}^{\mathit{free}(\xi)\setminus\mathsf{Reg}}$, Reg and $\mathbb{D}$ such that $L(\mathcal{A}) = L(\xi)$. Note that $\mathcal{A}$ can be considered as a register automaton over the alphabet $\Delta \times \{0,1\}^{\mathit{free}(\xi)}$ and $\mathbb{D}$. We construct a wRA $\mathcal{A}' = (\mathcal{L}, \mathsf{Reg}, \mathcal{L}_i, T, \mathcal{L}_f, \mathrm{wt}_{\mathrm{trans}}, \mathrm{wt}_{\mathsf{data}})$ over $\Delta\times\{0,1\}^{\mathit{free}(\xi)}$, $\mathbb{D}$ and $\mathbb{S}$ where $\mathrm{wt}_{\mathrm{trans}}$ and $\mathrm{wt}_{\mathsf{data}}$ are defined as follows. For any transition $t \in T$ with $\mathrm{label}(t) = (a,\iota,s,f_1,...,f_r)$, we let $\mathrm{wt}_{\mathrm{trans}}(t) = s$ and, for any register $Y_i \in \mathsf{Reg}$ ($i \in \{1,...,r\}$), we let $\mathrm{wt}_{\mathsf{data}}(t,Y_i) = f_i$. By Lemma 3.16 (1), we can construct a wRA $\mathcal{B}$ over $\Sigma \times \{0,1\}^{\mathit{free}(\xi)}$, $\mathbb{D}$ and $\mathbb{S}$ such that $[\![\mathcal{B}]\!] = \pi([\![\mathcal{A}']\!])$. Our goal now is to show that $[\![\mathcal{B}]\!] = [\![\zeta]\!]$. Indeed, let

$(w,\sigma) \in \mathbb{D}\Sigma^+_{free(\zeta)}$ with $w = (a_1,d_1)...(a_{|w|},d_{|w|})$. Then:

$$\begin{aligned}
&[\![\mathcal{B}]\!](w,\sigma)\\
&= \sum \left([\![\mathcal{A}']\!](w,u,v,\vec{g},\sigma) \mid u \in \{1,...,n\}^{|w|}, v \in (\tilde{S})^{|w|}, \vec{g} \in ((\tilde{\mathcal{F}})^{|w|})^r \right)\\
&= \sum \left(\prod_{j=1}^{|w|} v_j \cdot g_j^{(1)}(\delta_j^{(1)}, d_j) \cdot ... \cdot g_j^{(r)}(\delta_j^{(r)}, d_j) \;\middle|\; (w,u,v,\vec{g},\sigma) \in L(\mathcal{A}) \right)
\end{aligned}$$

where $\delta_j^{(1)}$, ..., $\delta_j^{(k)}$ are data stored in registers before taking the j-th transition (note that these data values are uniquely determined by $(w,u,v,\tilde{g},\sigma)$, since $\mathcal{A}$ is deterministic). Moreover, since $\mathcal{A}$ is a visibly register automaton, we have $g_j^{(k)}(\delta_j^{(k)}, d_j) = [\![g_j^{(k)}(Y_k,x)]\!](w,\sigma[x/j])$.
Since $L(\mathcal{A}) = L(\xi)$, we have

$$\begin{aligned}
&[\![\mathcal{B}]\!](w,\sigma)\\
&= \sum_{u \in \{1,...,n\}^{|w|}} \left(\prod_{j=1}^{|w|} v_j \cdot g_j^{(1)}(\delta_j^{(1)}, d_j) \cdot ... \cdot g_j^{(r)}(\delta_j^{(r)}, d_j) \;\middle|\; (w,u,v,\vec{f},\sigma) \models \xi \right)\\
&= \sum_{u \in \{1,...,n\}^{|w|}} \prod_{j=1}^{|w|} \left(s_{u(j)} \cdot \prod_{k=1}^{r} [\![f_{u(j),k}(Y_k,x)]\!](w,\sigma[x/j]) \;\middle|\; (w,\sigma[x/j]) \models \beta_{u(j)} \right).
\end{aligned}$$

One the other hand:

$$\begin{aligned}
&[\![\zeta]\!](w,\sigma)\\
&= \prod_{j=1}^{|w|} [\![\gamma]\!](w,\sigma[x/j])\\
&= \prod_{j=1}^{|w|} \sum_{i=1}^{n} \left(s_i \cdot \prod_{k=1}^{r} [\![f_{i,k}(Y_k,x)]\!](w,\sigma[x/j \;\middle|\; (w,\sigma[x/j]) \models \beta_i \right)\\
&= \sum_{u \in \{1,...,n\}^{|w|}} \prod_{j=1}^{|w|} \left(s_{u(j)} \cdot \prod_{k=1}^{r} [\![f_{u(j),k}(Y_k,x)]\!](w,\sigma[x/j]) \;\middle|\; (w,\sigma[x/j]) \models \beta_{u(j)} \right).
\end{aligned}$$

Then $[\![\mathcal{B}]\!](w,\sigma) = [\![\zeta]\!](w,\sigma)$ and hence the claim follows. □

The proof of Theorem 3.30 is by induction on the structure of a subformula ζ of ψ, applying Lemmas 3.31 - 3.34.

Proof of Theorem 3.21. Immediate by Theorems 3.28 and 3.30.

Chapter 4

Synchronizing Data Words for Register Automata

In this chapter, we study the concept of synchronizing data words in register automata: Does there exist a data word that sends all states of the given register automaton to a single state? The class of register automata that we consider here is the class of register automata of Chapter 3, but only over the data structure $\mathbb{D} = (\mathsf{D}, \{=_{\mathsf{D}}, \neq_{\mathsf{D}}\})$; they have a decidable non-emptiness problem [71], and the subclass of nondeterministic register automata with a single register has a decidable non-universality problem [36].

We introduce the notion of *data efficiency* for a data word; it gives the necessary and sufficient number of distinct data values of a synchronizing data word (if exists), to synchronize a register automaton. We show that the notion of data efficiency is tightly related to the complexity of deciding the existence of a synchronizing data word. For deterministic register automata with k registers (k-DRA), we prove that inputting data words with only $2k + 1$ distinct data values from the infinite data domain, is sufficient to synchronize. We show that the synchronizing problem for DRA is in general PSPACE-complete, and it is NLOGSPACE-membership for 1-DRA. For nondeterministic register automata (NRA), we show that Ackermann(n) distinct data (where n is the size of the given register automaton) might be necessary to synchronize. We establish Ackermann-completeness of the problem for 1-NRA, however, we show that the synchronizing problem for NRA is in general undecidable.

4.1 Synchronizing Data Words

In Chapter 3, to recognize the class of data languages, we considered the model of register automata over a data structure $\mathbb{D} = (\mathsf{D}, \mathcal{R})$, where we augmented a data domain D with some arbitrary set of binary relations $\mathcal{R}$ to handle the data comparison. Throughout all of this chapter, we only consider the class of register automata of Chapter 3, but only over the data structure $\mathbb{D} = (\mathsf{D}, \{=_{\mathsf{D}}, \neq_{\mathsf{D}}\})$. For the convenience of presentation, we avoid to use the notion of data structure $\mathbb{D}$, and instead we simply say data domain D.

For a data word $w = (a_1, d_1)(a_2, d_2)\dots(a_n, d_n) \in (\Sigma \times \mathsf{D})^+$, the length of w is $|w| = n$, and we use $\mathsf{data}(w) = \{d_1, \dots, d_n\} \subseteq \mathsf{D}$ to refer to the set of data values occurring in w, and we say that the *data efficiency of* w is $|\mathsf{data}(w)|$. As before, we let Reg be a finite set of registers, and we let $\nu : \mathsf{Reg} \to \mathsf{D}$ be a register valuation over Reg and D; recall that, sometimes we will consider $\nu = \begin{pmatrix} \nu(r_1) \\ \cdots \\ \nu(r_m) \end{pmatrix} \in \mathsf{D}^m$ where $\mathsf{Reg} = \{r_1, \cdots, r_m\}$, for $m \geqslant 1$. The register guards φ over Reg and $\{=_{\mathsf{D}}, \neq_{\mathsf{D}}\}$ are defined by the grammar

$$\varphi \ ::= \mathsf{True} \mid \ =r \mid \varphi \wedge \varphi \mid \neg\varphi,$$

where $r \in \mathsf{Reg}$. We simply use $\neq r$ for the inequality guard $\neg(=r)$. The satisfaction relation of register guards on $\mathsf{D}^m \times \mathsf{D}$ is defined as follows: (ν, d) satisfies the guard $= r$ iff $\nu(r) = d$; the other cases follow. For example, $(\begin{pmatrix} d_1 \\ d_2 \\ d_1 \end{pmatrix}, d_2)$ satisfies $((= r_1) \wedge (= r_2)) \vee (\neq r_3))$ where $d_1 \neq d_2$.

Since in this chapter we are interested in *synchronizing problem for register automata*, we consider register automata over an alphabet Σ and an infinite data domain D as a triple $\mathcal{A} = (\mathcal{L}, \mathsf{Reg}, T)$, i.e., without the finite sets of initial and final locations. This is due to the fact that the concept of synchronization requires that all runs of a register automaton, whatever the initial state (initial location and register valuations), end in the same state. As the data domains for registers are infinite, register automata are infinite-state transitions systems. We describe the behaviour of $\mathcal{A}$ as follows: Given the register automaton $\mathcal{A}$ in state $q = \langle \ell, \nu \rangle \in \mathcal{L} \times \mathsf{D}^{|\mathsf{Reg}|}$, while inputting the letter $a \in \Sigma$ and datum $d \in \mathsf{D}$, an a-transition $\ell \xrightarrow{\varphi,\, a,\, \mathsf{up}} \ell'$ may be fired if (ν, d) satisfies the guard φ; then $\mathcal{A}$ starts in *successor* state $q' = \langle \ell', \nu' \rangle$ where $\nu' = \nu[\mathsf{up} := d]$ is the update on registers. Recall that we write $\downarrow r$ when $\mathsf{up} = \{r\}$. By $\mathsf{post}(q, (a, d))$, we denote all successor states q' of q, on inputting letter a and datum d. A run of $\mathcal{A}$

over the data word $w = (a_1, d_1)(a_2, d_2) \cdots (a_n, d_n)$ is a sequence of states $q_0 q_1 \dots q_n$ where $q_i \in \mathsf{post}(q_{i-1}, (a_i, d_i))$ for all $1 \leqslant i \leqslant n$. We extend post to sets P of states by $\mathsf{post}(P, (a, d)) = \bigcup_{q \in P} \mathsf{post}(q, (a, d))$; and we extend post to data words by $\mathsf{post}(P, w \cdot (a, d)) = \mathsf{post}(\mathsf{post}(P, w), (a, d))$ for all data words $w \in (\Sigma \times \mathsf{D})^*$.

In the rest of this chapter, without loss of generality we consider *complete* register automata, meaning that for all states $q \in \mathcal{L} \times \mathsf{D}^{|\mathsf{Reg}|}$ and all inputs $(a, d) \in \Sigma \times \mathsf{D}$, there is at least one successor: $|\mathsf{post}(q, (a, d))| \geqslant 1$. We also classify the register automata into *deterministic* (DRA) and *nondeterministic* (NRA), where a register automaton is deterministic if $|\mathsf{post}(q, (a, d))| \leqslant 1$ for all states q and all inputs (a, d). The classes of deterministic register automata with a single register and with k registers are denoted by 1-DRA and k-DRA, respectively. The classes of nondeterministic register automata with a single register and with k registers are denoted by 1-NRA and $k -$ NRA, respectively.

The *synchronizing* problem for words is a well-studied concept for the class of deterministic finite state automata, or for short DFA; see [101]. The *deterministic finite-state automata* are tuples $\mathcal{B} = (Q, \Sigma, \Delta)$ where Q is a finite set of states, Σ is a finite alphabet and the transition function $\Delta : Q \times \Sigma \to Q$ is totally defined. The function Δ extends to finite words in a natural way, i.e., $\Delta(q, wa) = \Delta(\Delta(q, w), a)$ for all words $w \in \Sigma^*$ and letters $a \in \Sigma$; and it extends to all sets P of states by $\Delta(P, w) = \bigcup_{q \in P} \Delta(q, w)$. Informally, a synchronizing word leads the automaton from every state to the same state; formally speaking, the word $w \in \Sigma^*$ is synchronizing for $\mathcal{B} = (Q, \Sigma, \Delta)$ if there exists some state $\bar{q} \in Q$ such that $\Delta(Q, w) = \{\bar{q}\}$. The *synchronizing problem* in DFAs asks, given a DFA $\mathcal{A}$, whether there exists some synchronizing word for $\mathcal{A}$.

We introduce synchronizing data words for the class of register automata: for a register automaton $\mathcal{A} = (\mathcal{L}, \mathsf{Reg}, T)$ over an alphabet Σ and a data domain D, a data word $w \in (\Sigma \times \mathsf{D})^+$ is *synchronizing* if there exists some state $\langle \bar{\ell}, \bar{\nu} \rangle$ such that $\mathsf{post}(\mathcal{L} \times \mathsf{D}^{|\mathsf{Reg}|}, w) = \{\langle \bar{\ell}, \bar{\nu} \rangle\}$. The *synchronizing problem* asks, given a register automaton $\mathcal{A}$ over a data domain D, whether $\mathcal{A}$ has some synchronizing data word.

Synchronizing words have applications in planning, control of discrete event systems, biocomputing, and robotics [14, 101, 40]. To motivate our work, in Figure 4.1 we depict a web interface modelled by a register automaton $\mathcal{A}$ with register r, which models communications

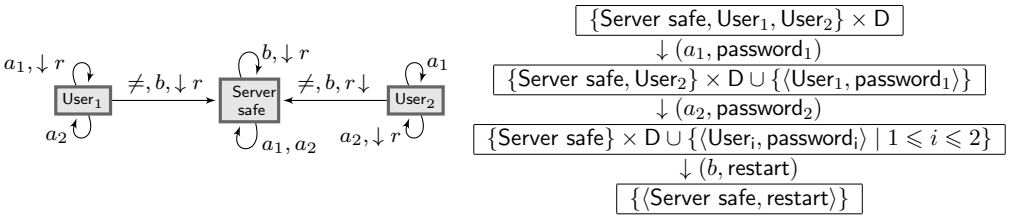


Figure 4.1: The interactive interfaces between a server and two users on the web.

between a server and two users over an interactive interface. The server execute commands a_1, a_2 or b, and users locally attach private information as data to the input. The register r in each user's interface can be used to store local information such as the password, which implies the server has only partial information about the current state of the users' interfaces. When the server detects that an attacker is eavesdropping on the communication, it guides the system to a *safe* state. The data word $w = (a_1, \mathsf{password}_1)(a_2, \mathsf{password}_2)(b, \mathsf{restart})$ with the distinct datum $\mathsf{restart}$, is synchronizing for $\mathcal{A}$. We display the successive states after reading each input of w in Figure 4.1. The computation starts in the infinite set of all states in which the server and users might be; registers may have stored any datum from the data domain D, ranging over *infinitely many possible data values* (e.g. ASCII strings or numbers). The input $(a_1, \mathsf{password}_1)$ updates r in interface of the user 1 which synchronizes the infinite set of states of that user in the state $(\mathsf{User}_1, \mathsf{password}_1)$. However, no update has taken place in interface of the user 2. In fact, the register of that interface may still store any datum from D; this changes after inputting $(a_2, \mathsf{password}_2)$. Using the last input $(b, \mathsf{restart})$, the server accomplishes synchronizing $\mathcal{A}$ into $(\mathsf{Server\ safe}, \mathsf{restart})$. Now, the users can renew their passwords to prevent the attacker from future eavesdropping.

4.2 Synchronizing Data Words for DRA

In this section, we establish the complexity bounds of the synchronizing problem for deterministic register automata with k registers (k-DRA). We prove that inputting words with data efficiency $2|\mathsf{Reg}| + 1$ is sufficient to synchronize a DRA. We then reduce the synchronizing problem for 1-DRA to the synchronizing problem in DFA, which yields NLOGSPACE-membership. We show that the problem for k-DRA, in general, can be decided in PSPACE; a reduction similar to the timed settings, as in [39],

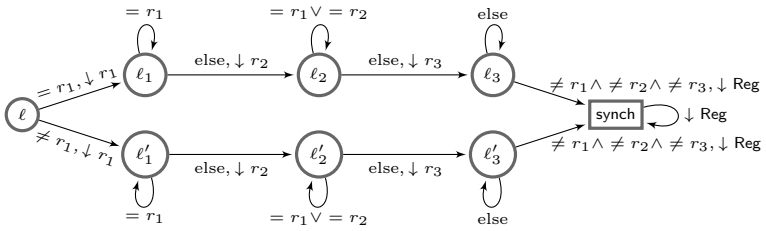


Figure 4.2: The DRA $\mathcal{A}$ of Example 4.1

provides the matching lower bound.

The concept of synchronization requires that all the runs of a register automaton end in the same state $\langle \ell_{\mathsf{synch}}, \nu_{\mathsf{synch}} \rangle$ that only depends on the data word w_{synch}, i.e., $\mathsf{post}(\mathcal{L} \times \mathsf{D}, w_{\mathsf{synch}}) = \{\langle \ell_{\mathsf{synch}}, \nu_{\mathsf{synch}} \rangle\}$. While processing a synchronizing data word, the infinite set of states in the automaton must necessarily be shrinked to a finite set of states. In Lemma 4.2, we prove that by means of data words with only $|\mathsf{Reg}|$ distinct data values we can shrink infinite states of a given register automaton $\mathcal{A} = (\mathcal{L}, \mathsf{Reg}, T)$ to a finite set. We establish this result based on the following two key facts:

1. To shrink the set $\mathcal{L} \times \mathsf{D}^{|\mathsf{Reg}|}$, for every $\ell \in \mathcal{L}$ one can find a word w_ℓ which brings $\mathcal{A}$ from the infinite set of state $\{\ell\} \times \mathsf{D}^{|\mathsf{Reg}|}$ to some finite set.

2. When reading a synchronizing data word w_{synch} from a state $\langle \ell, \nu \rangle$, some register r with $\nu(r) \in \mathsf{D} \backslash \mathsf{data}(w_{\mathsf{synch}})$ must be updated. Such updates must occur while taking inequality-guarded transitions which are accessible only by inequality-guarded transitions.

To make these explanations clear, we give the following example:

Example 4.1. *Consider the deterministic register automaton* $\mathcal{A} = (\mathcal{L}, \mathsf{Reg}, T)$ *over the single alphabet* $\Sigma = \{a\}$*, the data domain* D*, and the set of registers* $\mathsf{Reg} = \{r_1, r_2, r_3\}$ *depicted in Figure 4.2. Let* $\{d_1, d_2, d_3\} \subseteq \mathsf{D}$ *be a set of three distinct data values; One can easily see that for the data word* $w = (a, d_1)(a, d_2)(a, d_3)$ *we have* $\mathsf{post}(\{\ell\} \times \mathsf{D}^3, w) = \{\langle \ell_3, \begin{pmatrix} d_1 \\ d_2 \\ d_3 \end{pmatrix} \rangle, \langle \ell'_3, \begin{pmatrix} d_1 \\ d_2 \\ d_3 \end{pmatrix} \rangle\}$*. In addition, we can observe that* $\mathcal{A}$ *can be synchronized in the state* $\langle \mathsf{synch}, \begin{pmatrix} d_4 \\ d_4 \\ d_4 \end{pmatrix} \rangle$ *by the synchronizing data word* $w_{\mathsf{synch}} = (a, d_1)(a, d_2)(a, d_3)(a, d_4)$ *if* $\{d_1, d_2, d_3, d_4\} \subseteq \mathsf{D}$ *is a set of 4 distinct data.*

We use this observation to provide a linear bound on the sufficient number of required distinct data values while synchronizing register automata.

Lemma 4.2. *For all* DRA *for which there exist synchronizing data words, there exists some word w with data efficiency $|\mathsf{Reg}|$ such that $\mathsf{post}(\mathcal{L} \times \mathsf{D}^{|\mathsf{Reg}|}, w) \subseteq \mathcal{L} \times (\mathsf{data}(w))^{|\mathsf{Reg}|}$.*

Proof. Let $\mathcal{A} = (\mathcal{L}, \mathsf{Reg}, T)$ be k-DRA over the alphabet Σ and the data domain D, and let v be a synchronizing data word for $\mathcal{A}$. Let $k \geqslant 1$. We consider the case where $k < |\mathsf{data}(v)|$, else the statement of the lemma trivially holds.

For all $1 \leqslant i \leqslant |\mathsf{data}(v)|$, we say x_i is the i-th data appearing in $v = (a_1, d_1)(a_2, d_2) \cdots (a_n, d_n)$, if there exists $1 \leqslant j \leqslant |\mathsf{data}(v)|$ such that $x_i = d_j$, $x_i \notin \{d_1, \cdots, d_{j-1}\}$ and $|\{d_1, \cdots, d_j\}| = i$. Consider $\mathsf{val}_{\{x_1, \cdots, x_i\}} : \mathsf{Reg} \to \{x_1, \cdots, x_i\}$ as the register valuation over the first i data values in v. For the location $\ell \in \mathcal{L}$, the set $\mathsf{up} \subseteq \mathsf{Reg}$ of registers and the valuation $\mathsf{val}_{\{x_1, \cdots, x_i\}}$ we define a *symbolic state* as a tuple $(\ell, \mathsf{up}, \mathsf{val}_{\{x_1, \cdots, x_i\}})$ such that its semantics, denoted by $\langle\!\langle (\ell, \mathsf{up}, \mathsf{val}_{\{x_1, \cdots, x_i\}}) \rangle\!\rangle$ is defined as follows:

$$\langle\!\langle (\ell, \mathsf{up}, \mathsf{val}_{\{x_1, \cdots, x_i\}}) \rangle\!\rangle = \{\ell\} \times \{\nu \in \mathsf{D}^k \mid \nu(r) = \mathsf{val}(r) \text{ if } r \in \mathsf{up}\}.$$

Note that there are only finitely many symbolic states, each of them representing a set of states. To prove the statement of the lemma, we prove the following claim:

Claim 4.3. *For all locations $\hat{\ell} \in \mathcal{L}$ and for all $1 \leqslant i \leqslant k$, there exists some data word u_i such that*

- $\mathsf{data}(u_i) \subseteq \{x_1, x_2, \cdots, x_i\}$, *and*
- $\mathsf{post}(\{\hat{\ell}\} \times \mathsf{D}^k, u_i) \subseteq \bigcup_{\substack{\ell \in \mathcal{L}, \mathsf{up} \subseteq \mathsf{Reg} \\ |\mathsf{up}| \geqslant i}} \langle\!\langle (\ell, \mathsf{up}, \mathsf{val}_{\{x_1, \cdots, x_i\}}) \rangle\!\rangle$*; this means, all the states which are reached after reading u_i have at least i registers with values from $\{x_1, x_2, \cdots, x_i\}$.*

Proof of Claim 4.3. Let $\hat{\ell} \in \mathcal{L}$. We prove this claim by an induction on i.

Base of induction. Let $\mathsf{wait} = \{\hat{\ell}\} \times (\mathsf{D} \setminus \mathsf{data}(v))^k$ be the set of states with $\hat{\ell} \in \mathcal{L}$. Starting from states in wait, all runs of $\mathcal{A}$ over (a prefix of) v take the same sequence of the following transitions:

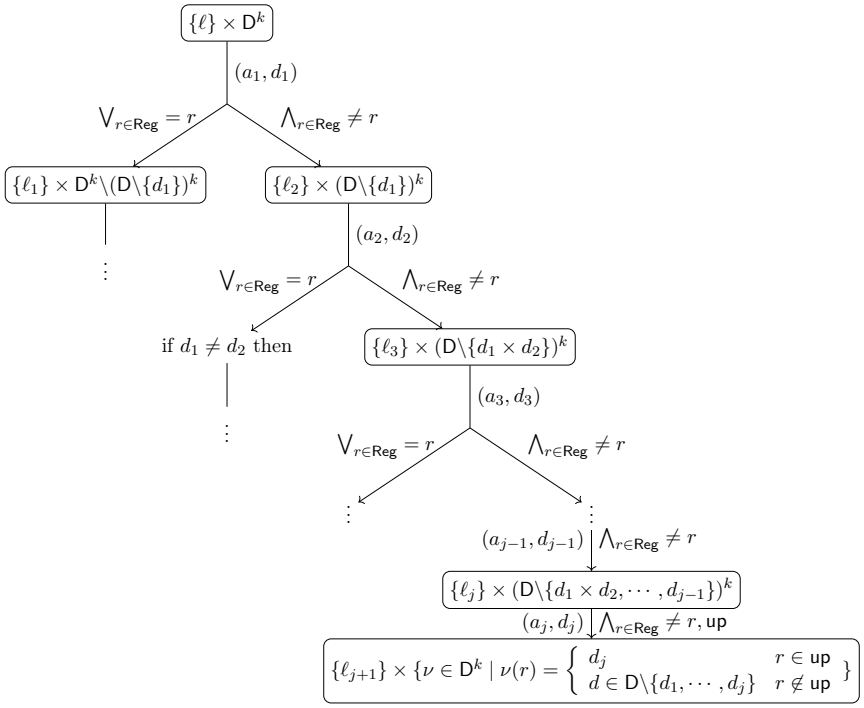


Figure 4.3: Runs of $\mathcal{A}$ over $(a_1, d_1)(a_2, d_2) \cdots (a_j, d_j)$

- a prefix of transitions $\xrightarrow{\bigwedge_{r\in\mathsf{Reg}} \neq r,\ \emptyset}$, i.e., transitions with inequality guards on all registers and with no register update,

- followed by a transition with inequality guard on all registers $\xrightarrow{\bigwedge_{r\in\mathsf{Reg}} \neq r,\ \mathsf{up}}$ updating registers in up, where $\emptyset \neq \mathsf{up} \subset \mathsf{Reg}$.

Otherwise, runs starting from any pairs of states $\langle \hat{\ell}, \nu_1 \rangle, \langle \hat{\ell}, \nu_2 \rangle \in \mathsf{wait}$ with $\nu_1 \neq \nu_2$ would end in some states $\langle \ell, \nu_1' \rangle, \langle \ell, \nu_2' \rangle \in \mathsf{wait}$ with $\nu_1' \neq \nu_2'$. This is a contradiction with the fact that the data word v is synchronizing data word.

Now, while reading the data word v, let the inequality-guarded transition $\xrightarrow{\bigwedge_{r\in\mathsf{Reg}} \neq r,\ \mathsf{up}}$ with $\mathsf{up} \neq \emptyset$, be taken by the j-th input (a_j, d_j); see Figure 4.3. We prove that the data word $u_1 = (a_1, x_1)(a_2, x_1) \cdots (a_j, x_1)$ with $\mathsf{data}(u_1) = \{x_1\}$ brings the set of states $\{\hat{\ell}\} \times \mathsf{D}^k$ to a subset in which

each state has some register with value x_1; in other words, we have

$$\mathsf{post}(\{\hat{\ell}\} \times \mathsf{D}^k, u_1) \subseteq \bigcup_{\substack{\ell \in \mathcal{L}, \mathsf{up} \subseteq \mathsf{Reg} \\ |\mathsf{up}| \geqslant 1}} \langle\!\langle (\ell, \mathsf{up}, \mathsf{val}_{\{x_1\}}) \rangle\!\rangle.$$

This phenomenon is depicted in Figure 4.4 and can be argued as follows. Observe that $x_1 = d_1$ is the first input data; thus after inputting (a_1, x_1) the set of successors is a disjoint union of two branches:

1. Either at least one register r has datum x_1 after taking the transition $\xrightarrow{\bigvee_{r \in \mathsf{Reg}} = r,\ a_1}$. By inputting $(a_2, x_1)(a_3, x_1) \cdots (a_j, x_1)$, all the followed successors of such states will always preserve the datum x_1 in the register r, even if some registers are updated;

2. or none of the registers is assigned to x_1 after taking the transition $\xrightarrow{\bigwedge_{r \in \mathsf{Reg}} \neq r,\ a_1}$. By inputting $(a_2, x_1)(a_3, x_1) \cdots (a_j, x_1)$, all the followed successors of such states, thus, take inequality-guarded transitions, and would not update any registers, unless for the last transition $\xrightarrow{\bigwedge_{r \in \mathsf{Reg}} \neq r,\ \mathsf{up}}$ fired by (a_j, x_1).

The above argument proves that u_1 with $\mathsf{data}(u_1) \subseteq \{x_1\}$ is the data word where, for every $\hat{\ell} \in \mathcal{L}$, we have

$$\mathsf{post}(\{\hat{\ell}\} \times \mathsf{D}^k, u_1) \subseteq \bigcup_{\substack{\ell \in \mathcal{L}, \mathsf{up} \subseteq \mathsf{Reg} \\ |\mathsf{up}| \geqslant 1}} \langle\!\langle (\ell, \mathsf{up}, \mathsf{val}_{\{x_1\}}) \rangle\!\rangle.$$

Thus the base of induction holds.

Step of induction. Assume that the induction hypothesis holds for $i-1$, namely, there exists some word u_{i-1} with $\mathsf{data}(u_{i-1}) \subseteq \{x_1, \cdots, x_{i-1}\}$ such that

$$\mathsf{post}(\{\hat{\ell}\} \times \mathsf{D}^k, u_{i-1}) \subseteq \bigcup_{\substack{\ell \in \mathcal{L}, \mathsf{up} \subseteq \mathsf{Reg} \\ |\mathsf{up}| \geqslant i-1}} \langle\!\langle (\ell, \mathsf{up}, \mathsf{val}_{\{x_1, \cdots, x_{i-1}\}}) \rangle\!\rangle.$$

We construct a data word $u_i = u_{i-1} \cdot u$ as follows: First, for each symbolic state $(\ell, \mathsf{up}, \mathsf{val}_{\{x_1, \cdots, x_{i-1}\}}) \in \mathsf{post}(\{\hat{\ell}\} \times \mathsf{D}^k, u_{i-1})$, reached after reading u_{i-1}, we construct a data word $u_{(\ell, \mathsf{up}, \mathsf{val}_{\{x_1, \cdots, x_{i-1}\}})}$ such that

- $\mathsf{data}(u_{(\ell, \mathsf{up}, \mathsf{val}_{\{x_1, \cdots, x_{i-1}\}})}) = \{x_1, x_2, \cdots, x_i\}$, and

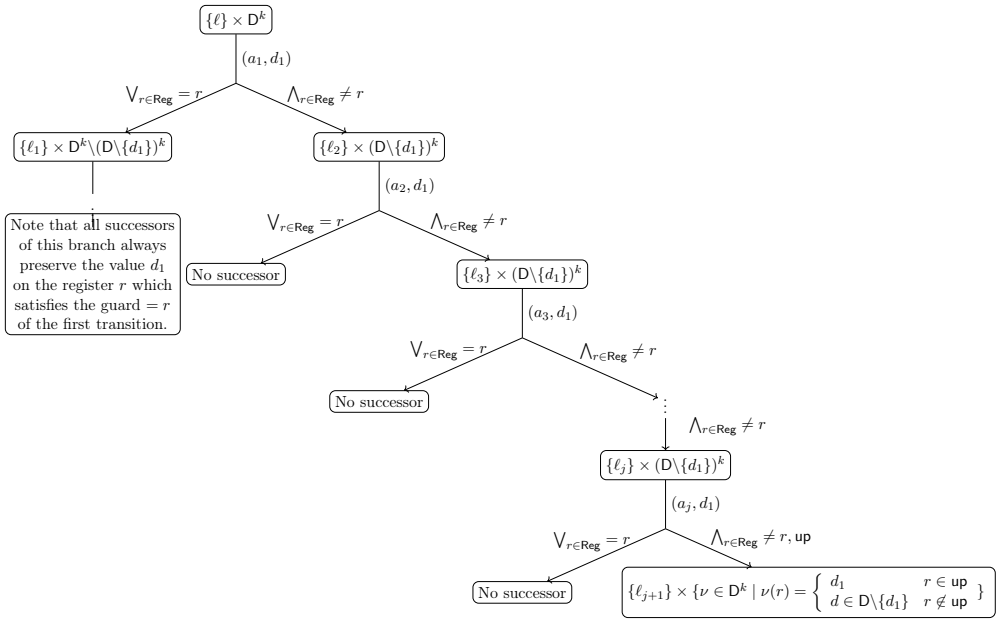


Figure 4.4: Runs of $\mathcal{A}$ over $u_1 = (a_1, d_1)(a_2, d_1)\cdots(a_j, d_1)$

- $\mathsf{post}(\langle\!\langle(\ell, \mathsf{up}, \mathsf{val}_{\{x_1,\cdots,x_{i-1}\}})\rangle\!\rangle, u_{(\ell,\mathsf{up},\mathsf{val}_{\{x_1,\cdots,x_{i-1}\}})}) \subseteq$
$$\bigcup_{\substack{\ell'\in\mathcal{L},\mathsf{up}'\subseteq\mathsf{Reg}\\ |\mathsf{up}'|\geqslant i}} \langle\!\langle(\ell', \mathsf{up}', \mathsf{val}_{\{x_1,\cdots,x_i\}})\rangle\!\rangle.$$

Then, the data word u, which appears as the suffix in the data word $u_i = u_{i-1} \cdot u$, is the concatenation of all such words $u_{(\ell,\mathsf{up},\mathsf{val}_{\{x_1,\cdots,x_{i-1}\}})}$ for each symbolic state $(\ell, \mathsf{up}, \mathsf{val}_{\{x_1,\cdots,x_{i-1}\}}) \in \mathsf{post}(\{\hat{\ell}\} \times \mathsf{D}^k, u_{i-1})$. Note that u satisfies the induction statement.

Now we construct the described data word $u_{(\ell,\mathsf{up},\mathsf{val}_{\{x_1,\cdots,x_{i-1}\}})}$ for a given symbolic state $\langle\!\langle(\ell, \mathsf{up}, \mathsf{val}_{\{x_1,\cdots,x_{i-1}\}})\rangle\!\rangle$. Without loss of generality, we assume that $|\mathsf{up}| = i - 1$; otherwise $u_{(\ell,\mathsf{up},\mathsf{val}_{\{x_1,\cdots,x_{i-1}\}})} = u_{i-1}$. Let

$$\mathsf{wait} = \langle\!\langle(\ell, \mathsf{up}, \mathsf{val}_{\{x_1,\cdots,x_{i-1}\}})\rangle\!\rangle \cap (\{\ell\} \times \{\nu \mid \nu(r) \in \mathsf{D} \setminus \mathsf{data}(v) \text{ if } r \notin \mathsf{up}\}).$$

All the runs of $\mathcal{A}$ over $v = (a_1, d_1)(a_2, d_2)\cdots(a_n, d_n)$ which start from the states in wait, take the same sequence of transitions. Since for all registers $r \in \mathsf{up}$ we have $\mathsf{val}_{\{x_1,\cdots,x_{i-1}\}}(r) \in \{x_1, x_2, \cdots, x_{i-1}\}$, after inputting successively data from $\mathsf{data}(v)$, all the reached states are also symbolic states. Let

- $(\ell^0, \mathsf{up}^0, \mathsf{val}^0_{\mathsf{data}(v)}) = (\ell, \mathsf{up}, \mathsf{val}_{\{x_1,\cdots,x_{i-1}\}})$, and
- $\mathsf{post}(\langle\!\langle(\ell^{j-1}, \mathsf{up}^{j-1}, \mathsf{val}^{j-1}_{\mathsf{data}(v)})\rangle\!\rangle, (a_j, d_j)) \subseteq \langle\!\langle(\ell^j, \mathsf{up}^j, \mathsf{val}^j_{\mathsf{data}(v)})\rangle\!\rangle$ for all $1 \leqslant j \leqslant n$.

In the sequel, we argue that there exists some $1 \leqslant m \leqslant n$ such that, in the sequence of transitions from one symbolic state to another symbolic state over the prefix $(a_1, d_1)(a_2, d_2)\cdots(a_m, d_m)$ of v (the first m inputs), the following holds:

1. For all $1 \leqslant j < m$, by inputting (a_j, d_j), we take the following transition

$$\langle\!\langle(\ell^{j-1}, \mathsf{up}^{j-1}, \mathsf{val}^{j-1}_{\mathsf{data}(v)})\rangle\!\rangle \xrightarrow{(\bigwedge_{r\in\Lambda_j} = r)\wedge(\bigwedge_{r\notin\Lambda_j} \neq r),\ a_j,\ \Gamma_j} \langle\!\langle(\ell^j, \mathsf{up}^j, \mathsf{val}^j_{\mathsf{data}(v)})\rangle\!\rangle$$

where $\Lambda_j, \Gamma_j \subseteq \mathsf{up}$. It implies that $\mathsf{val}^{j-1}_{\mathsf{data}(v)}(r) = d_j$ for all $r \in \Lambda_j$, and $\mathsf{val}^j_{\mathsf{data}(v)}(r) = d_j$ for all $r \in \Gamma_j$.

2. For $j = m$, by inputting (a_m, d_m), we take the following transition

$$\langle\!\langle(\ell^{m-1}, \mathsf{up}^{m-1}, \mathsf{val}^{m-1}_{\mathsf{data}(v)})\rangle\!\rangle \xrightarrow{(\bigwedge_{r\in\Lambda_m} = r)\wedge(\bigwedge_{r\notin\Lambda_m} \neq r),\ a_m,\ \Gamma_m} \langle\!\langle(\ell^m, \mathsf{up}^m, \mathsf{val}^m_{\mathsf{data}(v)})\rangle\!\rangle$$

where $\Lambda_m \subseteq \mathsf{up}$ wheras $\Gamma_m \not\subseteq \mathsf{up}$.

Now from the prefix $(a_1, d_1)(a_2, d_2)\cdots(a_m, d_m)$ of v, i.e., the first m inputs, and from the set of data $\{x_1, x_2, \cdots, x_i\}$, we construct the word $u_{\langle\!\langle(\ell,\mathsf{up},\mathsf{val}_{\{x_1,\cdots,x_{i-1}\}})\rangle\!\rangle} = (a_1, y_1)(a_2, y_2)\cdots(a_m, y_m)$ as follows: For all $1 \leqslant j \leqslant m$,

- if $\Lambda_j \neq \emptyset$, then there exists some register $r \in \mathsf{up}$ which stores the datum d_j, then $y_j = d_j$.
- if $\Lambda_j = \emptyset$, then none of the registers $r \in \mathsf{up}$ stores the datum d_j. Hence, $y_j = d$ where $d \in \{x_1, x_2, \cdots, x_i\} \setminus \{\mathsf{val}^{j-1}_{\mathsf{data}(v)}(r) \mid r \in \mathsf{up}\}$. The existence of such d is guaranteed since $|\mathsf{up}| = i - 1$ and $|\{x_1, x_2, \cdots, x_i\}| = i$. Moreover, since the transitions of the form $\xrightarrow{(\bigwedge_{r\in\mathsf{up}} \neq r),\ a_j\ \Gamma_j}$ have inequality guards for all registers, then changing the datum from d_j to y_j would result only in taking the same transition.

Note that the data values of the word $u_{(\ell,\mathsf{up},\mathsf{val}_{\{x_1,\cdots,x_{i-1}\}})}$ range over $\{x_1, \cdots, x_i\}$. As a result, all registers updated along the runs of $\mathcal{A}$

over $u_{(\ell,\mathsf{up},\mathsf{val}_{\{x_1,\cdots,x_{i-1}\}})}$ store some datum from $\{x_1,\cdots,x_i\}$. This argument shows that

$$\mathsf{post}(\langle\!\langle(\ell,\mathsf{up},\mathsf{val}_{\{x_1,\cdots,x_{i-1}\}})\rangle\!\rangle, u_{(\ell,\mathsf{up},\mathsf{val}_{\{x_1,\cdots,x_{i-1}\}})}) \subseteq \bigcup_{\substack{\ell'\in\mathcal{L},\mathsf{up}'\subseteq\mathsf{Reg}\\ |\mathsf{up}'|\geqslant i}} \langle\!\langle(\ell',\mathsf{up}',\mathsf{val}_{\{x_1,\cdots,x_i\}})\rangle\!\rangle.$$

Since the word $u_i = u_{i-1}\cdot u$, where u is the concatenation of all such words $u_{(\ell,\mathsf{up},\mathsf{val}_{\{x_1,\cdots,x_{i-1}\}})}$, then the step of induction holds.

We have proved by induction that for all locations $\hat{\ell}\in\mathcal{L}$ and all $1\leqslant i\leqslant k$, there exists a data word u_i such that

- $\mathsf{data}(u_i)\subseteq\{x_1,x_2,\cdots,x_i\}$, and
- $\mathsf{post}(\{\hat{\ell}\}\times\mathsf{D}^k,u_i)\subseteq\bigcup_{\substack{\ell\in\mathcal{L},\mathsf{up}\subseteq\mathsf{Reg}\\ |\mathsf{up}|\geqslant i}}\langle\!\langle(\ell,\mathsf{up},\mathsf{val}_{\{x_1,\cdots,x_i\}})\rangle\!\rangle$.

It completes the proof for the Claim 4.3. □

For the location $\ell\in\mathcal{L}$, let $w_\ell = u_k$ be the data word which satisfies the conditions stated in Claim 4.3. Then, the data word $w=(w_\ell)_{\ell\in\mathcal{L}}$ proves the statement of Lemma 4.2.

□

After reading some word that shrinks the infinite set of states in a register automaton to a finite set P, one can apply the *pairwise synchronization* technique to synchronize states in P. This technique is the core to decide the synchronizing problem for DFA is in NLOGSPACE: Given a DFA $\mathcal{A}=(Q,\Sigma,\Delta)$, it is known that $\mathcal{A}$ has a synchronizing word if and only if for all pairs of states $q,q'\in Q$, there exists a word v such that $\Delta(q,v)=\Delta(q',v)$ (see [101] for more details). The pairwise synchronization then let $P_n=Q$, and for all $i=|Q|-1,\cdots,1$ repeats the following: find a word v_i such that $\Delta(q,v_i)=\Delta(q',v_i)$ for some pair $q,q'\in P_{i+1}$ and let $P_i=\Delta(P_{i+1},v_i)$. The word $w=v_{n-1}\cdots v_2\cdot v_1$ is synchronizing for the DFA. We generalize the pairwise synchronization technique for DRA to establish the following Lemma.

Lemma 4.4. *For all* DRA *with some synchronizing data words, there exists a synchronizing word with data efficiency* $2|\mathsf{Reg}|+1$.

Proof. Let $\mathcal{A}=(\mathcal{L},\mathsf{Reg},T)$ be a k-DRA over the alphabet Σ and the data domain D. Let $k\geqslant 1$. Recall that for a data word w we denote by $\mathsf{data}(w)$ the data values which appear in w. Here, for every state $q=\langle\ell,\nu\rangle$ we use the same notation to define $\mathsf{data}(q)=\{\nu(r)\mid r\in\mathsf{Reg}\}$ which denotes the

data appearing in the register valuation ν of q.

Let $\pi : Y_1 \to Y_2$ be a bijection on data values where $Y_1, Y_2 \subseteq \mathsf{D}$. For every state $q = \langle \ell, \nu \rangle$ define $\pi(q) = \langle \ell, \nu' \rangle$, where ν' satisfies $\nu'(r) = \pi(\nu(r))$ for all $r \in \mathsf{Reg}$. Now for every data word $w = (a_1, d_1) \cdots (a_n, d_n)$, we define $\pi(w) = (a_1, \pi(d_1)) \cdots (a_n, \pi(d_n))$. Note that the application of π on q preserves the reachability property, i.e., $\mathsf{post}(\pi(q), \pi(w)) = \{\pi(q') \mid q' \in \mathsf{post}(q, w)\}$.

With the assumption that $\mathcal{A}$ has some synchronizing data word, we first prove the following claim by induction.

Claim 4.5. *For all pairs of states q_1, q_2, if there exists a data word w such that $|\mathsf{post}(\{q_1, q_2\}, w)| = 1$, then for all sets $X = \{x_1, x_2, \cdots, x_{2k+1}\} \subset \mathsf{D}$ where $\mathsf{data}(q_1), \mathsf{data}(q_2) \subseteq X$, there exists some data word $w_{q_1,q_2} \in (\Sigma \times X)^*$ such that $|\mathsf{post}(\{q_1, q_2\}, w_{q_1,q_2})| = 1$. Note that since $|X| = 2k+1$, the data efficiency of w_{q_1,q_2} is at most $2k+1$.*

Proof of Claim 4.5. Let q_1 and q_2 be two states of $\mathcal{A}$ and let $\mathsf{data}(q_1, q_2) = \mathsf{data}(q_1) \cup \mathsf{data}(q_2)$. By the assumption that $\mathcal{A}$ has some synchronizing data word, we know that there exists a data word w such that $|\mathsf{post}(\{q_1, q_2\}, w)| = 1$. The proof is by an induction on the length of w.

Base of induction. Assume $w = (a, d)$, and so $|w| = 1$. Let X be any arbitrary set of data valuse such that $|X| = 2k + 1$ and $\mathsf{data}(q_1, q_2) \subseteq X$. We consider two possible cases:

1. $d \in X$. This requires that $\mathsf{data}(w) \subseteq X$. Hence, $w_{q_1,q_2} = w$ satisfies the induction statement.

2. $d \notin X$. Since $|\mathsf{data}(q_1, q_2)| \leqslant 2k$, there exists a data value x, $x \neq d$, such that $x = X \setminus \mathsf{data}(q_1, q_2)$. Since $x \neq d$, we can define the bijection $\pi : \{d\} \cup \mathsf{data}(q_1, q_2) \to \{x\} \cup \mathsf{data}(q_1, q_2)$ such that $\pi(d) = x$ and $\pi(d') = d'$ for all $d' \in \mathsf{data}(q_1, q_2)$. Observe that $\pi(q_i) = q_i$ for all $i \in \{1, 2\}$. Then,

$$\begin{aligned} |\mathsf{post}(\{q_1, q_2\}, (a, d))| &= |\mathsf{post}(\{\pi(q_1), \pi(q_2)\}, (a, \pi(d)))| \\ &= |\mathsf{post}(\{q_1, q_2\}, (a, x))|. \end{aligned}$$

From this and the assumption that $|\mathsf{post}(\{q_1, q_2\}, (a, d)| = 1$ we obtain $|\mathsf{post}(\{q_1, q_2\}, (a, x))| = 1$. Thus the data word $w_{q_1,q_2} = (a, x)$ satisfies the

induction statement. It mean base of induction holds

Step of induction. Suppose that the induction hypothesis holds for $i-1$. Let $(a,d)\cdot w$ be a data word such that $|w| = i-1$ and $|\mathsf{post}(\{q_1,q_2\},(a,d)\cdot w)| = 1$. Let $X \subset \mathsf{D}$ with $|X| = 2k+1$ and $\mathsf{data}(q_1,q_2) \subseteq X$. We construct the data word w_{q_1,q_2} as follows. Let $p_1 = \mathsf{post}(q_1,(a,d))$ and $p_2 = \mathsf{post}(q_2,(a,d))$, and let $\mathsf{data}(p_1,p_2) = \mathsf{data}(p_1)\cup\mathsf{data}(p_2)$. Since p_1,p_2 are successors of q_1,q_2 after reading (a,d), we know that if $d \in \mathsf{data}(q_1,q_2)$ then $d \in \mathsf{data}(p_1,p_2)$. We consider two possible cases:

1. $d \in \mathsf{data}(q_1,q_2)$ or $d \notin \mathsf{data}(p_1,p_2)$. If $d \in \mathsf{data}(q_1,q_2)$, then obviously $\mathsf{data}(p_1,p_2) \subseteq \mathsf{data}(q_1,q_2)$. If $d \notin \mathsf{data}(p_1,p_2)$, then $\mathsf{data}(p_1,p_2) = \mathsf{data}(q_1,q_2)$. As a result, $\mathsf{data}(p_1,p_2) \subseteq X$. By induction hypothesis, there exists some data word w_{p_1,p_2} over data domain $X \subset \mathsf{D}$ such that $|\mathsf{post}(\{p_1,p_2\},w_{p_1,p_2})| = 1$. For $w_{q_1,q_2} = (a,d)\cdot w_{p_1,p_2}$ the statement of induction holds, as $|\mathsf{post}(\{q_1,q_2\},w_{q_1,q_2})| = 1$.

2. $d \notin \mathsf{data}(q_1,q_2)$ and $d \in \mathsf{data}(p_1,p_2)$. Without loss of generality, we assume that $d \notin X$. Otherwise $d \in X$ would imply $\mathsf{data}(p_1,p_2) \subseteq X$, and we would simply let $w_{q_1,q_2} = w_{p_1,p_2}$. Since $|\mathsf{data}(q_1,q_2)| \leqslant 2k$, there exists data $x \neq d$ such that $x = X \setminus \mathsf{data}(q_1,q_2)$. Since $x \neq d$, we can define the bijection $\pi : \{d\}\cup\mathsf{data}(q_1,q_2) \to \{x\}\cup\mathsf{data}(q_1,q_2)$ such that $\pi(d) = x$ and $\pi(d') = d'$ for all $d' \in \mathsf{data}(q_1,q_2)$. Since $\mathsf{data}(p_1,p_2)\setminus\{d\} \subseteq \mathsf{data}(q_1,q_2)$, having d in the domain of π, the bijection π ranges over $\mathsf{data}(p_1,p_2)$. By induction hypothesis, there exists some data word w_{p_1,p_2} over data domain $(X\setminus\{x\})\cup(\{d\})$ such that $|\mathsf{post}(\{p_1,p_2\},w_{p_1,p_2})| = 1$. Then, $|\mathsf{post}(\{\pi(p_1),\pi(p_2)\},\pi(w_{p_1,p_2}))| = 1$. For every $i \in \{1,2\}$, we have $\pi(p_i) \in \mathsf{post}(q_i,(a,x))$, since $p_i \in \mathsf{post}(q_i,(a,d))$ and $x = \pi(d)$. By above arguments, we conclude that $|\mathsf{post}((\{q_1,q_2\},(a,x)\pi(w_{p_1,p_2})| = 1$. As $\{x\}\cup\mathsf{data}(q_1,q_2) \subseteq X$, thus the data word $w_{q_1,q_2} = (a,x)\pi(w_{p_1,p_2})$ satisfies the statement of induction.

So far, we have proved that there exists $w_{q_1,q_2} \in (\Sigma\times X)^*$ that brings every two states q_1 and q_2 into a singleton, which completes the proof of Claim 4.5.

□

By the assumption that $\mathcal{A}$ has some synchronizing data word, and by using Lemma 4.2, we know that there exists some word w with data efficiency k such that $\mathsf{post}(\mathcal{L}\times\mathsf{D}^k,w) \subseteq \mathcal{L}\times\mathsf{data}(w)^k$. Now consider a

set $X = \{x_1, x_2, \cdots, x_{2k+1}\} \subset \mathsf{D}$ such that $\mathsf{data}(w) \subseteq X$. We use the pairwise synchronization technique as follows: Define $P_n = \mathcal{L} \times X^k$ and $n = |\mathcal{L}|(2k+1)^k$ which implies $|P_n| = n$. For all $i = n-1, \cdots, 1$ repeat the following:

- Take a pair of states $q_1, q_2 \in P_{i+1}$. By Claim 4.5, one can find some word $w_{q_1,q_2} \in (\Sigma \times X)^*$ such that $|\mathsf{post}(\{q_1, q_2\}, w_{q_1,q_2})| = 1$,
- Let $v_i = w_{q_1,q_2}$ and $P_i = \mathsf{post}(P_{i+1}, v_i)$.

Note that by determinism of $\mathcal{A}$, for every $i \in \{1, \cdots, n-1\}$, we have $|P_i| \leqslant |P_{i+1}| - 1$. Thus the word $w_{\mathsf{synch}} = w \cdot v_{n-1} \cdots v_2 \cdot v_1$ is a synchronizing data word for $\mathcal{A}$. Since $\mathsf{data}(w) \subseteq X$ and $\mathsf{data}(v_i) \subseteq X$ for every $i \in \{1, \cdots, n-1\}$, thus the data efficiency of w_{synch} is at most $2k+1$. This completes the proof of Lemma 4.4.

□

Given a 1-DRA $\mathcal{A}$, the synchronizing problem can be solved as follows. First, one needs to ensure that from each location ℓ an update on the single register is achieved by going through inequality-guarded transitions, which can be done in NLOGSPACE. Lemma 4.2 suggests that feeding $\mathcal{A}$ consecutively with a single datum $x \in \mathsf{D}$ is sufficient for this step and the set of successors of $\mathcal{L} \times D$ would be a subset of $\mathcal{L} \times \{x\}$. Then, by picking an arbitrary set $\{x, y, z\}$ of data including x, using Lemma 4.4 and the pairwise synchronization technique, the problem can be reduced to the synchronizing problem in DFA where the data stored in registers and the input data extend the locations and the alphabet: $Q = \mathcal{L} \times \{x, y, z\}$ and $\Sigma \times \{x, y, z\}$.

In the following lemma, we provide a family of k-DRA with a single alphabet $\Sigma = \{a\}$, for which the linear bound on data efficiency of synchronizing data words is necessary. This sufficient bound is crucial to establish membership of synchronizing k-DRA in PSPACE.

Lemma 4.6. *There is a family of* DRA $(\mathcal{A}_n)_{n \in \mathbb{N}}$*, with* $n = |\mathsf{Reg}|$ *registers,* $\mathcal{O}(n)$ *locations and single alphabet* $\Sigma = \{a\}$*, such that all synchronizing data words have data efficiency* $\mathcal{O}(n)$*.*

Proof. The family of register automata $\mathcal{A}_n (n \in \mathbb{N})$ is defined over an alphabet $\Sigma = \{a\}$, and an infinite data domain D. Each register automaton $\mathcal{A}_n$ has n registers and has a single letter a. The construction of $\mathcal{A}_n$ is based on two distinguished locations ℓ_{init} and synch, and two connected chains of distinguished locations, each of them contains n locations $\ell_1, \ell_2, \cdots, \ell_n$ and $\ell'_1, \ell'_2, \cdots, \ell'_n$, respectively. The register automaton depicted in Figure 4.2

shows $\mathcal{A}_3$ in this family. The only transition in synch is a self-loop with update on all n registers. This means $\mathcal{A}_n$ can only be synchronized in synch. There are two transitions leaving ℓ_{init}, each of them going to one of the chains:

$$\ell_{\mathsf{init}} \xrightarrow{=r_1,\, a,\, \downarrow r_1} \ell_1 \qquad \text{and} \qquad \ell_{\mathsf{init}} \xrightarrow{\neq r_1,\, a,\, \downarrow r_1} \ell'_1.$$

Therefore, $\mathsf{post}(\{\ell_{\mathsf{init}}\} \times \mathsf{D}^n, (a, x)) = \{\ell_1, \ell'_1\} \times (\{x\} \times \mathsf{D}^{n-1})$ for all $x \in \mathsf{D}$. Generally speaking, from $\{\ell_1, \ell'_1\} \times (\{x\} \times \mathsf{D}^{n-1})$ the i-th locations in both chains are simultaneously reached after inputting i distinct data values. Formally, for all $1 \leqslant i < n$, from each location ℓ_i and ℓ'_i there are two possible transitions. One transition is a self-loop with the guard $\bigvee_{r \in \{r_1, \cdots, r_i\}} = r_i$, where $r_1, \cdots, r_i$ are those registers which have been updated so far. The other goes to the next location in the corresponding chain, storing the fresh input datum in register r_{i+1}. Namely, the following transitions:

$$\ell_i \xrightarrow{\bigvee_{r \in \{r_1, \cdots, r_i\}} (= r_i),\ a} \ell_i \qquad \text{and} \qquad \ell_i \xrightarrow{\mathsf{else},\, a,\, \downarrow r_{i+1}} \ell_{i+1},$$

$$\ell'_i \xrightarrow{\bigvee_{r \in \{r_1, \cdots, r_i\}} (= r_i),\ a} \ell'_i \qquad \text{and} \qquad \ell'_i \xrightarrow{\mathsf{else},\, a,\, \downarrow r_{i+1}} \ell'_{i+1}.$$

On the last locations ℓ_n and ℓ'_n of each chain, one transition with inequality guards and update on all registers leaves the chain to synch. Otherwise the successor is the location, itself. Namely, the following transitions:

$$\ell_n \xrightarrow{\bigwedge_{r \in \mathsf{Reg}} (\neq r_i),\, a,\, \mathsf{Reg}} \mathsf{synch} \qquad \text{and} \qquad \ell_n \xrightarrow{\mathsf{else},\, a} \ell_n,$$

$$\ell'_n \xrightarrow{\bigwedge_{r \in \mathsf{Reg}} (\neq r_i),\, a,\, \mathsf{Reg}} \mathsf{synch} \qquad \text{and} \qquad \ell'_n \xrightarrow{\mathsf{else},\, a} \ell'_n.$$

By the given construction, we can see that, from the infinite set $\{\ell_{\mathsf{init}}\} \times \mathsf{D}^n$, $n+1$ distinct data values must be read to synchronize $\mathcal{A}_n$ in the location synch. Since $\mathcal{A}_n$ can only be synchronized in synch, all synchronizing data words then have data efficiency at least $n + 1 \in O(n)$.

All remains is to prove that $\mathcal{A}_n$ has indeed some synchronizing data word. For this, let $\{x_1, x_2, \cdots, x_{n+1}\}$ be a set of $n+1$ distinct data values and $w_{\mathsf{synch}} = (a, x_1)(a, x_2) \cdots (a, x_n)(a, x_{n+1})$. For the the whole set of space $\mathcal{L} = \{\ell_{\mathsf{init}}, \mathsf{synch}, \ell_1, \ell_2, \cdots, \ell_n, \ell'_1, \ell'_2, \cdots, \ell'_n\}$, one can easily observe that $\mathsf{post}(\mathcal{L} \times \mathsf{D}^n, w_{\mathsf{synch}}) = \{\langle \mathsf{synch}, x_{n+1} \rangle\}$, and $|\mathsf{data}(w_{\mathsf{synch}})| = n+1$. □

Theorem 4.7. *The synchronizing problem for k-*Dra *is* PSPACE*-complete.*

Proof. For PSPACE-hardness, we adapt an established reduction (see, e.g., [39]) from the non-emptiness problem for k-DRA. The result then follows by PSPACE-completeness of the non-emptiness problem for k-DRA [36].

Let $\mathcal{A} = (\mathcal{L}, \mathsf{Reg}, T)$ be a k-DRA over Σ and D. We equip $\mathcal{A}$ with an initial location ℓ_i and an accepting location ℓ_f. Without loss of generality, we assume that all outgoing transitions from ℓ_i update all registers, and that ℓ_f has no outgoing transition. We also assume that $\mathcal{A}$ is complete; otherwise, we add some non-accepting location and connect all undefined transitions to it. The reduction is in a way that from $\mathcal{A}$ we construct another k-DRA $\mathcal{A}_{\mathsf{syn}}$ such that the language of $\mathcal{A}$ is not empty if, and only if, $\mathcal{A}_{\mathsf{syn}}$ has some synchronizing data word. We define $\mathcal{A}_{\mathsf{syn}} = (\mathcal{L}_{\mathsf{syn}}, \mathsf{Reg}, T_{\mathsf{syn}})$ over the alphabet Σ_{syn} and the data domain D as follows. The set of locations is $\mathcal{L}_{\mathsf{syn}} = \mathcal{L} \cup \{\mathsf{reset}\}$, where $\mathsf{reset} \notin \mathcal{L}$ is a new location; the alphabet $\Sigma_{\mathsf{syn}} = \Sigma \cup \{\star\}$, where $\star \notin \Sigma$. To define T_{syn}, we add the following transitions to T.

- $\ell_f \xrightarrow{a,\, \mathsf{Reg}} \ell_f$ for all letters $a \in \Sigma_{\mathsf{syn}}$,
- $\ell_i \xrightarrow{\star,\, \mathsf{Reg}} \ell_i$
- $\mathsf{reset} \xrightarrow{a,\, \mathsf{Reg}} \ell_i$ for all letters $a \in \Sigma_{\mathsf{syn}}$,
- $\ell \xrightarrow{\star,\, \mathsf{Reg}} \mathsf{reset}$ for all $\ell \in \mathcal{L}_{\mathsf{syn}}$ except for $\mathsf{reset}, \ell_i, \ell_f$.

Note that $\mathcal{A}_{\mathsf{synch}}$ is indeed deterministic and complete. To establish the correctness of the reduction, we prove that the language of $\mathcal{A}$ is not empty if, and only if, $\mathcal{A}_{\mathsf{syn}}$ has a synchronizing data word. First, assume that the language of $\mathcal{A}$ is not empty. Then there exists a data word $w = (a_1, d_1) \dots (a_n, d_n)$ such that $w \in L(\mathcal{A})$. Hence there exists a run starting from $\langle \ell_i, \nu_i \rangle$ and ending in $\langle \ell_f, \nu_f \rangle$ for some $\nu_i, \nu_f \in \mathsf{D}^{|\mathsf{Reg}|}$. The data word $(\star, d)(\star, d) w (\star, d)$ for some $d \in \mathsf{D}$ synchronizes $\mathcal{A}_{\mathsf{syn}}$ in location ℓ_f. Second, assume that $\mathcal{A}_{\mathsf{syn}}$ has some synchronizing data word. Let w be one of the shortest data synchronizing data words. All transitions in ℓ_f are self-loops with update on all registers; Hence, $\mathcal{A}_{\mathsf{syn}}$ can only be synchronized in ℓ_f. Therefore, we also have $\mathsf{post}(\langle \ell_i, \nu_i \rangle, w) = \{\langle \ell_f, \nu_f \rangle\}$ (for some $\nu_i, \nu_f \in \mathsf{D}^{|\mathsf{Reg}|}$). By the fact that w is a shortest synchronizing data word, we can infer that the corresponding run does not contain any $\star$-transitions except for two self-loops in ℓ_i in the very beginning. □

4.3 Synchronizing Data Words for NRA

In this section, we study the synchronizing problems for NRA. We update a result in [39] to present a general reduction from the *non-universality* problem to the synchronizing problem in NRA. This reduction proves the *undecidability* of the synchronizing problem in k-NRA, and $\mathsf{Ackermann}$-hardness in 1-NRA. We then prove that in 1-NRA the synchronizing and non-universality problems are indeed interreducible, which completes the picture by $\mathsf{Ackermann}$-completeness of the synchronizing problem in 1-NRA.

While synchronizing in 1-NRA (with $\mathcal{O}(n)$ locations), we show that we can implement two kinds of counting features. For this, we introduce two families of 1-NRA: A family where an input datum $x \in \mathsf{D}$ must be read 2^n times, and another family where $\mathsf{Ackermann}(n)$ distinct data must be read to achieve synchronization. We give the construction of the two families in the following lemmas. We say an x-token is in location ℓ after reading a data word v if $\langle \ell, x \rangle \in \mathsf{post}(\mathcal{L} \times \mathsf{D}, v)$.

Lemma 4.8. *There is a family of 1-NRA $(\mathcal{A}_{\mathsf{counter}(n)})_{n\in\mathbb{N}}$ with $\mathcal{O}(n)$ locations such that for all synchronizing data words w, some datum $d \in \mathsf{data}(w)$ appears in w at least 2^n times.*

Proof. We define the family of 1-NRAs $(\mathcal{A}_{\mathsf{counter}(n)})_{n\in\mathbb{N}}$ over an infinite data domain D. The alphabet of each register automata $\mathcal{A}_{\mathsf{counter}(n)}$ is $\Sigma = \{\#, \star, \mathsf{Bit}_0, \mathsf{Bit}_1, \cdots, \mathsf{Bit}_n\}$. The construction of $\mathcal{A}_{\mathsf{counter}(n)}$ is based on three distinguished locations $\mathsf{synch}, \mathsf{reset}, \mathsf{zero}$ and locations $2^n, 2^{n-1}, \cdots, 2^1, 2^0$ and $2^n_c, 2^{n-1}_c, \cdots, 2^1_c, 2^0_c$. The register automaton $\mathcal{A}_{\mathsf{counter}(n)}$, depicted in Figure 4.5, is designed in a way that for all synchronizing data words w, some datum $d \in \mathsf{data}(w)$ appears in w at least 2^n times. A counting feature is thus embedded in $\mathcal{A}_{\mathsf{counter}(n)}$; intuitively, the set of all reached states represents the counter value. Starting from $\{\langle \mathsf{zero}, d\rangle\}$, the first increment results in $\{2^n_c, \cdots, 2^2_c, 2^1_c, 2^0\} \times \{d\}$, where location 2^i means that the i-th least significant bit in binary representation of the counter value is set to $\mathtt{1}$, and location 2^i_c means that the i-th bit is set to $\mathtt{0}$. Informally, we say that there is an d-token in every reached location, i.e., $2^n_c, \cdots, 2^2_c, 2^1_c, 2^0$ have d-tokens. Increments are encoded by replacing d-tokens in the following sets of locations:

$$\{2^n_c, \cdots, 2^2_c, 2^1, 2^0_c\},\ \{2^n_c, \cdots, 2^2_c, 2^1, 2^0\},\ \{2^n_c, \cdots, 2^3_c, 2^2, 2^1_c, 2^0_c\},\ \text{etc.}$$

The transitions of $\mathcal{A}_{\mathsf{counter}(n)}$ are defined in such a way that, starting from $\{(\mathsf{zero}, d)\}$, *either* 2^i have tokens *or* 2^i_c, but *never* both of them at the same

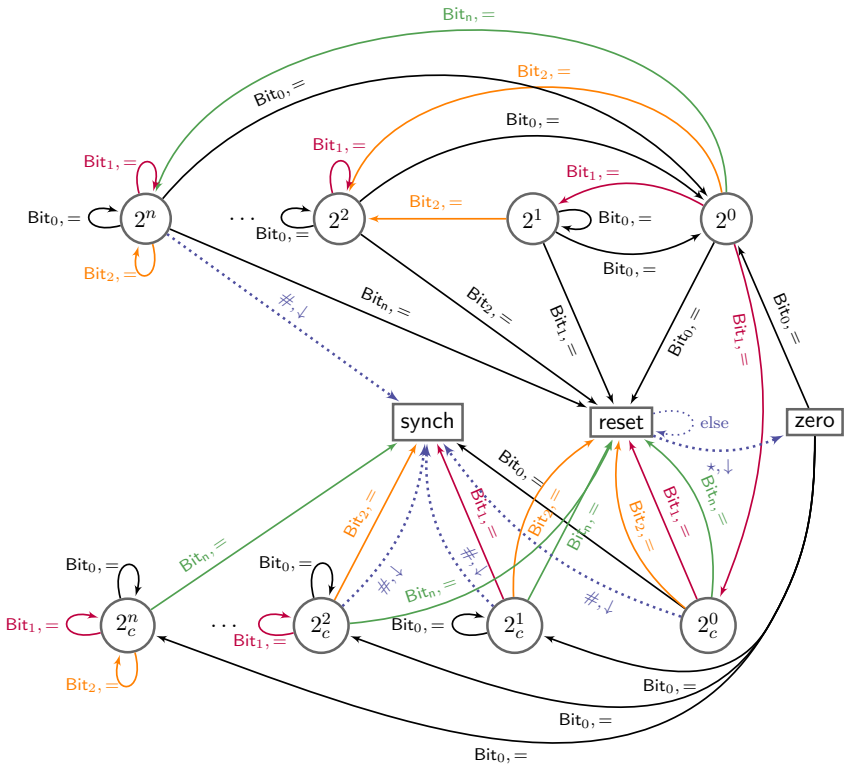


Figure 4.5: The 1-NRA $\mathcal{A}_{\mathsf{counter}(n)}$ implementing a binary counter.

time. All transitions in synch are self-loops with update on the register, i.e., synch $\xrightarrow{\Sigma,\downarrow r}$ synch; thus, $\mathcal{A}_{\mathsf{counter}(n)}$ can only be synchronized in synch. Moreover, synch is only accessible with #-transitions; assuming w is one of the shortest synchronizing words, we see that $\mathsf{post}(\mathcal{L}\times\mathsf{D}, w) = \{\langle \mathsf{synch}, d\rangle)\}$ where w ends with $(\#, d)$. The counting involves *resetting* and *incrementing* processes.

1. ***resetting* to zero.** The $\star$-transitions are devised to place a token in zero, i.e., from all locations $\ell \in \mathcal{L} \setminus \{\mathsf{synch}\}$ we have $\ell \xrightarrow{\star,\downarrow r}$ zero. All synchronizing words read $\star$ at least once; otherwise those runs which start from reset would never be synchronized. All *incorrect transitions* with the incrementing process are the ones directed to reset to enforce another $\star$ and to reset the counter to 0.

2. ***incrementing process.*** We use $\mathsf{Bit}_0, \ldots, \mathsf{Bit}_n$-transitions with equality guards to control the increment. Intuitively, a Bit_i-transition is taken to set the i-th bit of binary representation of the counter value according to the standard rules of binary incrementation. Initially, by following the transitions $\mathsf{zero} \xrightarrow{=r,\ \mathsf{Bit}_0} 2^0$ and $\mathsf{zero} \xrightarrow{=r,\ \mathsf{Bit}_0} 2^j_c$ for all $1 \leqslant j \leqslant n$, the d-token in zero splits in 2^0 and $2^n_c, \cdots 2^1_c$ to represent $\mathtt{0}\cdots\mathtt{01}$. Equality-guarded Bit_i-transitions, for $i \in \{1, \ldots, n\}$, are incorrect transitions from the location zero. Whenever a datum different from d is processed, $\mathcal{A}_{\mathsf{counter}(n)}$ takes self-loops (omitted in Figure 4.5) and keeps the d-tokens unmoved. The Bit_i-transitions should only be taken if the i-th bit is not set, i.e., if 2^i has no token. This is guaranteed by the Bit_i-transition $2^i \xrightarrow{=r,\ \mathsf{Bit}_i} \mathsf{reset}$, which results in an *incorrect transition*. Note that such a transition must be avoided; otherwise the counting process has to restart from $\mathtt{0}$. In addition, we need to guarantee that a Bit_i-transition, for all $i \geqslant 1$, is taken only if all less significant bits are set; in other words, if all locations $2^{i-1}, \cdots 2^0$ have tokens. For testing whether all locations $2^{i-1}, \cdots 2^0$ have tokens, we use the Bit_i-transitions in $2^{i-1}_c, \cdots 2^0_c$, i.e., $2^j_c \xrightarrow{=r,\ \mathsf{Bit}_i} \mathsf{reset}$ for all $j \leqslant i$. Moreover, Bit_i-transitions must produce tokens in 2^i and $2^0_c, \cdots 2^{i-1}_c$. Thus the transitions $2^j \xrightarrow{=r,\ \mathsf{Bit}_i} 2^i$ and $2^j \xrightarrow{=r,\ \mathsf{Bit}_i} 2^j_c$ for all $j \leqslant i$ will be taken. We also need to place tokens in 2^0 whenever the counter value represents an even number. A Bit_0-transitions produces an d-token in 2^0. The counter value is even if, and only if, 2^0_c has a token; $\mathsf{Bit}_1, \cdots, \mathsf{Bit}_n$-transitions, i.e., the transitions $2^0_c \xrightarrow{=r,\ \mathsf{Bit}_j} \mathsf{reset}$ for all $0 < j \leqslant n$, should be avoided in this case; taking such transitions will result in an incorrect simulation and thus should be avoided. It is important that the Bit_i-transitions (for $0 \leqslant i \leqslant n$) also consume the token in 2^i_c, thus $2^i_c \xrightarrow{=r,\ \mathsf{Bit}_i} \mathsf{synch}$.

By this construction, it is easy to see that Bit_i-transitions are the only way to produce a token in 2^i, which can be fired if 2^i_c has an d-token. The Bit_i-transitions then consume the d-token in 2^i_c. This guarantees that after the first *reset* to zero, the two locations 2^i and 2^i_c would never have a token at the same time. Finally, the #-transition in 2^n_c and all locations 2^i (with $i < n$) are directed to reset. As we can expect, in contrast, the #-transition in 2^n and all locations 2^i_c (with $i < n$) are directed to synch. This guarantees that the counter must once correctly count from $\mathtt{0}$ to $\mathtt{10}\cdots\mathtt{0}$, meaning that at least one datum d appears at least 2^n times while synchronizing $\mathcal{A}_{\mathsf{counter}(n)}$. □

Next, we show that the data efficiency while synchronizing 1-NRA can be a function in the *fast growing hierarchy* [89], which we refer as data-counting feature. We let $\mathsf{tower} : \mathbb{N} \to \mathbb{N}$ be defined inductively by $\mathsf{tower}(0) = 1$ and $\mathsf{tower}(n+1) = 2^{\mathsf{tower}(n)}$. We prove the following lemma.

Lemma 4.9. *There is a family of* 1*-NRA* $(\mathcal{A}_{\mathsf{tower}(n)})_{n\in\mathbb{N}}$ *with* $O(n)$ *locations, such that* $|\mathsf{data}(w)| \in \mathcal{O}(\mathsf{tower}(n))$ *for all synchronizing data words* w*.*

Proof. The family of 1-NRAs $(\mathcal{A}_{\mathsf{tower}(n)})_{n\in\mathbb{N}}$ is defined over the natural numbers $\mathbb{N}$. The alphabet of $\mathcal{A}_{\mathsf{tower}(n)}$ is $\Sigma = \{\#, \star, a, \mathsf{rep}, \mathsf{doub}, \mathsf{tow}\}$. The construction of$\mathcal{A}_{\mathsf{tower}(n)}$ is based on n locations $\mathsf{Data}_1, \mathsf{Data}_{1,2}, \cdots, \mathsf{Data}_{1,2,\cdots,n}$ and 6 more locations $\mathsf{synch}, \mathsf{reset}, \mathsf{waitRep}, \mathsf{waitDoub}, \mathsf{Rep}, \mathsf{store}$. The $\mathcal{A}_{\mathsf{tower}}$ depicted in Figure 4.6 is indeed $\mathcal{A}_{\mathsf{tower}(3)}$ from this family; $\mathcal{A}_{\mathsf{tower}(n)}$ has, in general, the same structure as $\mathcal{A}_{\mathsf{tower}(3)}$, only with a chain of n locations $\mathsf{Data}_1, \mathsf{Data}_{1,2}, \cdots, \mathsf{Data}_{1,2,\cdots,n}$. The register automaton $\mathcal{A}_{\mathsf{tower}(n)}$ is constructed in a way that for all synchronizing data words$w$ we have $|\mathsf{data}(w)| \in O(\mathsf{tower}(n))$. All transitions in synch are self-loops with an update on the register, i.e., the slef-loops $\mathsf{synch} \xrightarrow{\Sigma,\downarrow r} \mathsf{synch}$; thus, $\mathcal{A}_{\mathsf{tower}(n)}$ is only synchronized in synch. Moreover, synch is only accessible with #-transitions; assuming w is one of the shortest synchronizing words, we see that $\mathsf{post}(\mathcal{L} \times \mathsf{D}, w) = \{\langle \mathsf{synch}, d\rangle\}$ where w ends with $(\#, d)$. From all locations $\ell \in \mathcal{L} \setminus \{\mathsf{synch}\}$ we have $\ell \xrightarrow{\star,\downarrow r} \mathsf{Data}_1$; we say that $\star$-transitions *reset* $\mathcal{A}_{\mathsf{tower}(n)}$. Moreover, the only outgoing transition in location reset is the $\star$-transition. Thus, a *reset* must happen while synchronizing. After this forced reset, say on reading $(\star, 1)$, the set of reached states is $\{\langle \mathsf{Data}_1, 1\rangle, \langle \mathsf{synch}, 1\rangle\}$. When $\langle \ell, d\rangle$ is in the set of reached states, we say an d-token is in ℓ; for example, the forced reset places a 1-token in Data_1 and synch each. Since resetting is inefficient, we try to avoid it; All transitions which are lead to reset are called *inefficient*.

Since the question is the required data efficiency of synchronizing words, we always start from datum 1 and feed $\mathcal{A}_{\mathsf{tower}}$ with the smallest number i which results in synchronizing. Moreover, while *resetting* we read datum 1. For all locations $\mathsf{Data}_{1,\cdots,i}$ with $1 \leqslant i < n$, then we have the following transitions:

$$\mathsf{Data}_{1,\cdots,i} \xrightarrow{\neq r,\ \mathsf{rep}} \mathsf{Data}_{1,\cdots,i+1} \quad \text{and} \quad \mathsf{Data}_{1,\cdots,i} \xrightarrow{\neq r,\ \mathsf{rep},\downarrow r} \mathsf{Data}_{1,\cdots,i+1}.$$

All other transitions in $\mathsf{Data}_{1,\cdots,i}$ are inefficient. Below, we rename the location $\mathsf{Data}_{1,2,\cdots,n}$ to $\mathsf{waitTow}$. All transitions in $\mathsf{waitTow}$, $\mathsf{waitDoub}$, $\mathsf{waitRep}$, Rep and store are as in $\mathcal{A}_{\mathsf{tower}}$ depicted in Figure 4.6. Except on the following transitions:

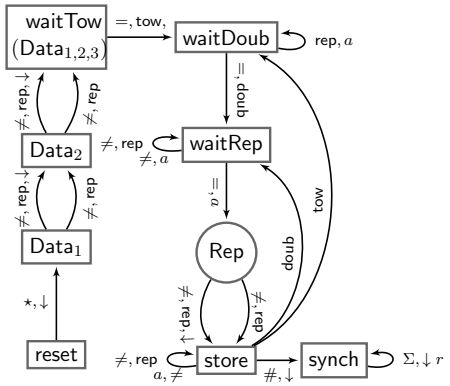


Figure 4.6: $\mathcal{A}_{\mathsf{tower}}$: $\mathsf{tower}(3)$ distinct data must be read to synchronize.

1. $\mathsf{waitTow} \xrightarrow{=r,\ \mathsf{tow}} \mathsf{waitDoub}$.
2. $\mathsf{waitDoub} \xrightarrow{=r,\ \mathsf{doub}} \mathsf{waitRep}$, $\mathsf{waitDoub} \xrightarrow{a} \mathsf{waitDoub}$ and $\mathsf{waitDoub} \xrightarrow{\mathsf{rep}} \mathsf{waitDoub}$.
3. $\mathsf{waitRep} \xrightarrow{=r,\ a} \mathsf{Rep}$, $\mathsf{waitRep} \xrightarrow{\neq r,\ a} \mathsf{waitRep}$ and $\mathsf{waitRep} \xrightarrow{\neq r,\ \mathsf{rep}} \mathsf{waitRep}$,
4. $\mathsf{Rep} \xrightarrow{\neq r,\ \mathsf{rep}} \mathsf{store}$ and $\mathsf{Rep} \xrightarrow{\neq r,\ \mathsf{rep},\ \downarrow r} \mathsf{store}$,
5. $\mathsf{store} \xrightarrow{\mathsf{tow}} \mathsf{waitDoub}$, $\mathsf{store} \xrightarrow{\mathsf{doub}} \mathsf{waitRep}$, $\mathsf{store} \xrightarrow{\neq r,\ a} \mathsf{store}$ and $\mathsf{store} \xrightarrow{\neq r,\ \mathsf{rep}} \mathsf{store}$.

Note that $\mathsf{store} \xrightarrow{\#,\ \downarrow r} \mathsf{synch}$, and all the other transitions are inefficient.

The crucial point is that while synchronizing $\mathcal{A}_{\mathsf{tower}}$, some inequality-guarded transitions are inescapable, which are the ones that may *replicate* the tokens. For example, if there is one token in Data_1, firing two transitions $\mathsf{Data}_1 \xrightarrow{\neq r,\ \mathsf{rep}} \mathsf{Data}_{1,2}$ and $\mathsf{Data}_1 \xrightarrow{\neq r,\ \mathsf{rep},\ \downarrow r} \mathsf{Data}_{1,2}$ replicates it to two tokens in $\mathsf{Data}_{1,2}$. After the (forced) reset, having a 1-token in Data_1, the only word that results in synchronizing is $(\mathsf{rep}, 2)(\mathsf{rep}, 3)\cdots(\mathsf{rep}, n)$. It replicates the 1-token to n tokens: $\{1, 2, \cdots, n\}$-tokens, and places them in $\mathsf{waitTow}$. The only efficient transition is then $\mathsf{waitTow} \xrightarrow{=r,\ \mathsf{tow}} \mathsf{waitDoub}$. In particular, the #-transition activates a reset; as a result, as long as some token is in $\mathsf{waitTow}$, #-transitions must be avoided. This implies that for all $1 \leqslant i \leqslant n$, the i-token in $\mathsf{waitTow}$ can leave the location, only individually, on inputting (tow, i). We say that inputting (tow, i) releases the

$\mathsf{tower}(i)$-process. Intuitively, the i-token in $\mathsf{waitTow}$ is waiting to releases the $\mathsf{tower}(i)$-process, right after the process of $\mathsf{tower}(i-1)$ is accomplished. By an induction on i, we prove that releasing a $\mathsf{tower}(i)$-process results in $\mathsf{tower}(i)$ tokens in store: $\{1, 2, 3, \cdots, \mathsf{tower}(i)\}$-tokens.

Base of induction. The $\mathsf{tower}(1)$-process starts with moving the 1-token from $\mathsf{waitTow}$ to $\mathsf{waitDoub}$. The only efficient transitions are then fired by $(\mathsf{doub}, 1)(a, 1)(\mathsf{rep}, 2)$, which split the 1-token to $\{1, 2\}$-tokens in store.

Step of induction. Assume that the $\mathsf{tower}(i-1)$-process has produced $\{1, 2, 3, \cdots, \mathsf{tower}(i-1)\}$-tokens in store. The $\mathsf{tower}(i)$-process starts with the only efficient transitions, which are tow-transitions. They are the transitions which move the i-token to $\mathsf{waitTow}$ and which move all $\{1, 2, 3, \cdots, \mathsf{tower}(i-1)\}$-tokens from store to $\mathsf{waitDoub}$. Recall that $\mathsf{tower}(i) = 2^{\mathsf{tower}(i-1)}$, simply $\mathsf{tower}(i-1)$ times doubling 1. Each i-token waiting in $\mathsf{waitDoub}$ is supposed to release a doubling:

- 1-token: the only efficient transitions are on $(\mathsf{doub}, 1)(a, 1)(\mathsf{rep}, 2)$, which replicate $\{1, 2\}$-tokens in store.
- 2-token: the only efficient transition is on $(\mathsf{doub}, 2)$, which (also) brings the result of the first doubling into $\mathsf{waitRep}$. Both $\{1, 2\}$-tokens in $\mathsf{waitRep}$ then will be replicated individually. Note that while replicating, a locally fresh datum from all data in $\mathsf{waitRep}, \mathsf{Rep}$ and store must be read; otherwise an inefficient transition is fired. The $\{1, 2, 3, 4\}$-tokens are produced in store, as a result of the second doubling, by $(a, 1)(\mathsf{rep}, 3)(a, 2)(\mathsf{rep}, 4)$.
- j-token: inputting (doub, j) brings the result of the previous doubling, i.e., $\{1, 2, \cdots, 2^{j-1}\}$-tokens, into $\mathsf{waitRep}$. For all $1 \leqslant m \leqslant 2^{j-1}$, the m-token is replicated into $\{m, 2^{j-1}+m\}$-tokens by $(a, m)(\mathsf{rep}, 2^{j-1} + m)$. This computation produces all $\{1, 2, \cdots, 2^j\}$-tokens in store.
- $\mathsf{tower}(i)$-token: it doubles the number of tokens in store for the $\mathsf{tower}(i-1)$-th time, and thus $\{1, 2, 3, \cdots, \mathsf{tower}(i)\}$-tokens, where $\mathsf{tower}(i) = 2^{\mathsf{tower}(i-1)}$, are produced in store.

The above argument proves the step of induction.

After the $\mathsf{tower}(n)$-process, the #-transition synchronizes the register automaton $\mathcal{A}_{\mathsf{tower}(n)}$ in the location synch. Note that for all $1 \leqslant i \leqslant n$, the $\mathsf{tower}(i)$-process is forced due to the equality guard on

the only efficient transition $\mathsf{waitTow} \xrightarrow{=r,\ \mathsf{tow}} \mathsf{waitDoub}$. As a result, we proved that, after the $\mathsf{tower}(n)$-process, $\mathsf{tower}(n)$ distinct tokens (distinct data values) are produced in store, which implies $|\mathsf{data}(w)| \in O(\mathsf{tower}(n))$ for all synchronizing words w. □

Remark 4.10. *Recall that, from [89], that* tower *is at level* 3 *of the Ackermann-hierarchy. Using similar ideas as in Lemma 4.9, we can define a family of* 1*-NRA* $\mathcal{A}_n^m$ $(n, m \in \mathbb{N})$ *such that all synchronizing data words have data efficiency at least* $\mathsf{ack}_n(m)$*, where* ack_n *is at level* n *of the Ackermann-hierarchy.*

To define the language of a given register automaton $\mathcal{A}$, we equip it with an initial location ℓ_i and a set $\mathcal{L}_f$ of accepting locations, where, without loss of generality, we assume that all outgoing transitions from ℓ_i update all registers. The language $L(\mathcal{A})$ is the set of all data words $w \in (\Sigma \times \mathsf{D})^+$, for which there is a run from $\langle \ell_i, \nu_i \rangle$ to $\langle \ell_f, \nu_f \rangle$ such that $\ell_f \in \mathcal{L}_f$ and $\nu_i, \nu_f \in \mathsf{D}^{|\mathsf{Reg}|}$. The universality problem asks, given an RA, whether $L(\mathcal{A}) = (\Sigma \times \mathsf{D})^+$. The aim is to prove the undecidability result for the synchronizing problem in k-NRA, i.e., the following theorem:

Theorem 4.11. *The synchronizing problem for* k*-*NRA *is undecidable.*

To prove Theorem 4.11, first we adopt an established reduction in [39] to provide the following lemma.

Lemma 4.12. *The non-universality problem is reducible to the synchronizing problem for* NRA*.*

Proof. Let $\mathcal{A} = (\mathcal{L}, \mathsf{Reg}, T)$ be an NRA over Σ and D, equipped with an initial location ℓ_i and a set $\mathcal{L}_f$ of accepting locations, where, without loss of generality, we assume that all outgoing transitions from ℓ_i update all registers. We also assume that $\mathcal{A}$ is complete, otherwise, we add some non-accepting location and direct all undefined transitions to it. The reduction is in a way that from $\mathcal{A}$ we construct another register automaton $\mathcal{A}_{\mathsf{syn}}$ such that the language of $\mathcal{A}$ is not universal if and only if $\mathcal{A}_{\mathsf{syn}}$ has some synchronizing data word. We define $\mathcal{A}_{\mathsf{syn}} = (\mathcal{L}_{\mathsf{syn}}, \mathsf{Reg}, T_{\mathsf{syn}})$ over Σ_{syn} and D as follows. The set of locations is $\mathcal{L}_{\mathsf{syn}} = \mathcal{L} \cup \{\mathsf{reset}, \mathsf{synch}\}$ where $\mathsf{synch}, \mathsf{reset} \notin \mathcal{L}$ are two new locations; and the alphabet is $\Sigma_{\mathsf{synch}} = \Sigma \cup \{\#, \star\}$ where $\#, \star \notin \Sigma$. To define T_{syn}, we add the following transitions to T:

- $\mathsf{synch} \xrightarrow{a,\ \mathsf{Reg}} \mathsf{synch}$ for all letters $a \in \Sigma_{\mathsf{syn}}$,
- $\mathsf{reset} \xrightarrow{\star,\ \mathsf{Reg}} \ell_i$ and $\mathsf{reset} \xrightarrow{a,\ \mathsf{Reg}} \mathsf{reset}$ for all letters $a \in \Sigma_{\mathsf{syn}} \backslash \{\star\}$,

- $\ell \xrightarrow{\star,\,\mathsf{Reg}} \ell_i$ for all locations $\ell \in \mathcal{L}$,
- $\ell \xrightarrow{\#,\,\mathsf{Reg}} \mathsf{synch}$ for all non-accepting locations $\ell \in \mathcal{L} \backslash \mathcal{L}_f$,
- $\ell \xrightarrow{\#,\,\mathsf{Reg}} \mathsf{reset}$ for all accepting locations $\ell \in \mathcal{L}_f$.

To establish the correctness of the reduction, we prove that the language of $\mathcal{A}$ is not universal if and only if $\mathcal{A}_{\mathsf{syn}}$ has a synchronizing data word.

First, assume that the language of $\mathcal{A}$ is not universal. Then there exists a word $w = (a_1, d_1) \dots (a_n, d_n)$ such that $w \notin L(\mathcal{A})$. Hence, all runs starting in $\langle \ell_i, \nu_i \rangle$ with $\nu_i \in \mathsf{D}^{|\mathsf{Reg}|}$, end in some state $\langle \ell, \nu \rangle$ with $\ell \notin \mathcal{L}_f$. The data word $(\star, d) \cdot w \cdot (\#, d)$ with $d \in \mathsf{D}$ synchronizes $\mathcal{A}_{\mathsf{syn}}$ in location synch, proving that $\mathcal{A}_{\mathsf{syn}}$ has some synchronizing data word.

Second, assume that $\mathcal{A}_{\mathsf{syn}}$ has some synchronizing data word. All transitions in synch are self-loops with update on all registers; thus, $\mathcal{A}_{\mathsf{syn}}$ is only synchronized in synch. Moreover, synch is only accessible with $\#$-transitions; assuming w is one of the shortest synchronizing data words, we see that $\mathsf{post}(\mathcal{L} \times \mathsf{D}, w) = \{\langle \mathsf{synch}, \nu \rangle\}$ for some $\nu \in \mathsf{D}^{|\mathsf{Reg}|}$. From all locations $\ell \in \mathcal{L}$ we have $\ell \xrightarrow{\star,\,\mathsf{Reg}} \ell_i$; we say that $\star$-transitions *reset* $\mathcal{A}_{\mathsf{syn}}$. Moreover, the only outgoing transition in location reset is the $\star$-transition. Thus, a *reset* followed by some $\#$ must occur while synchronizing. Let $w = w_0(\star, d_\star)w_1(\#, d_\#)w_2$, where $w_1 \in (\Sigma \times \mathsf{D})^+$ is the data word between the last occurrence of $\star$ and the first following occurrence of $\#$, and $w_2 \in (\Sigma' \backslash \{\star\})^*$. We prove that $w_1 \notin L(\mathcal{A})$ and hence $L(\mathcal{A}) \neq (\Sigma \times \mathsf{D})^+$. By contradiction, assume that w_1 is in the language; thus, there exist valuations $\nu_i, \nu_f \in \mathsf{D}^{|\mathsf{Reg}|}$ such that $\mathcal{A}_{\mathsf{syn}}$ has a run over w_1, *i.e.*, starting in $\langle \ell_i, \nu_i \rangle$ and ending in $\langle \ell_f, \nu_f \rangle$ where $\ell_f \in \mathcal{L}_f$. In fact, since all outgoing transitions in ℓ_i update all registers, then for all valuations ν_i, $\mathcal{A}_{\mathsf{syn}}$ has an accepting run over w_1.

Note that w_0 cannot be a synchronizing word for $\mathcal{A}_{\mathsf{syn}}$, because this would contradict the assumption that w is one of the shortest synchronizing data word. It implies that there must be some state q such that $\mathsf{post}_{\mathcal{A}_{\mathsf{syn}}}(q, w_0)$ contains some state $\langle \ell, \nu \rangle$ with $\ell \neq \mathsf{synch}$. From $\langle \ell, \nu \rangle$, inputting the next $(\star, d_\star)$ (that is after w_0 in synchronizing word w), we reach $\langle \ell_i, \{d_\star\}^{|\mathsf{Reg}|} \rangle$. Since for all valuations ν_i, starting in $\langle \ell_i, \nu_i \rangle$, $\mathcal{A}_{\mathsf{synch}}$ has an accepting run over w_1, it must have an accepting run from $\langle \ell_i, \{d_\star\}^{|\mathsf{Reg}|} \rangle$ to some accepting state $\langle \ell_f, \nu_f \rangle$ too. Reading the last $\#$ (that is after w_1 in synchronizing word w), reset is reached. Since w_2 does not contain any

$\star$, reset is never left, meaning that $\mathcal{A}_{\mathsf{syn}}$ cannot synchronize in synch, a contradiction. The proof is complete. □

The proof of Theorem 4.11 is an immediate result of Lemma 4.12 and the undecidability of the non-universality problem for k-NRA (Theorems 2.7 and 5.4 in [36]).

Bellow, we present a reduction which shows that, for 1-NRA, the synchronizing problem is reducible to the non-universality problem. It provides the tight complexity bounds for the synchronizing problem.

Lemma 4.13. *The synchronizing problem is reducible to the non-universality problem for 1-*NRA*.*

Proof. We establish a reduction from the synchronizing problem to the non-universality problem in 1-NRA as follows. Given a register automaton $\mathcal{A} = (\mathcal{L}, \mathsf{Reg}, T)$ over Σ and D, we construct another register automaton $\mathcal{A}_{\mathsf{comp}}$ equipped with an initial location and a set of final locations such that $\mathcal{A}$ has some synchronizing words if and only if the language of $\mathcal{A}_{\mathsf{comp}}$ is not universal.

First, we observe that Lemma 4.2 holds for 1-NRA, i.e., for all 1-NRA with some synchronizing data word, there exists some word w with data efficiency 1 such that $\mathsf{post}(\mathcal{L} \times \mathsf{D}, w) \subseteq \mathcal{L} \times \mathsf{data}(w)$. For all locations $\ell \in \mathcal{L}$, such words must update the register by firing an inequality-guarded transition, that is reached only via inequality-guarded transitions; this can be checked in $\mathsf{NLOGSPACE}$. Given $\mathcal{A}$, we assume that such word w always exists; otherwise, we define $\mathcal{A}_{\mathsf{comp}}$ to be a register automaton with a single accepting state with self-loops on all letters, and thus the language of $\mathcal{A}_{\mathsf{comp}}$ is universal. Let $\mathsf{data}(w) = \{x\}$, we then say that $\mathcal{A}$ has some synchronizing word v if $\mathsf{post}(\mathcal{L} \times \{x\}, v)$ is a singleton.

Second, we define a data language lang such that data words in this language are encodings of the synchronizing process. Let $\mathcal{L} = \{\ell_1, \ell_2, \cdots, \ell_n\}$ be the set of locations and x, y two distinct data. Each data word in lang, if there exists any,

- starts with the *initial block*: containing a delimiter $(\star, y)$, the sequence $(\ell_1, x), (\ell_2, x), \cdots, (\ell_n, x)$ and an input $(a, d) \in \Sigma \times \mathsf{D}$ as the beginning of a synchronizing word.
- follows with *normal blocks*: containing the delimiter $(\star, y)$, successors reached from states and input of the previous block, and the next input of the synchronizing word.

- ends with the *final block*: containing the delimiter $(\star, y)$, a single successor reached from states and input of the previous block and the delimiter $(\star, y)$.

The language lang is over the alphabet $\Sigma_{\mathsf{lang}} = \Sigma \cup \mathcal{L} \cup \{\star\}$ where $\star \notin \Sigma \cup \mathcal{L}$. The language $\mathsf{lang} \subseteq (\Sigma_{\mathsf{lang}} \times \mathsf{D})^+$ is the set of all data words u that satisfy all following *membership conditions*:

1. The data words u starts with $(\star, y)(\ell_1, x), (\ell_2, x), \cdots, (\ell_n, x)$ for some $x, y \in \mathsf{D}$ with $y \neq x$; this condition guarantees the correctness of the encoding for the initial block.

2. Let $\mathsf{proj}(u)$ be the projection of u into Σ_{lang} (i.e., omitting the data values). Then there exists some $\ell_{\mathsf{synch}} \in \mathcal{L}$ where $\mathsf{proj}(u) \in (\star\, \mathcal{L}^+ \Sigma)^+ \star\, \ell_{\mathsf{synch}} \star$. This condition guarantees the right form of data words as the encodings of synchronizing processes.

The next two conditions guarantee the uniqueness of the delimiter:

3. The letter $\star$ in u occurs only with datum y.

4. No other letter in u occurs with datum y.

The next three conditions guarantee that all the successors that can be reached from states and inputs in each block are correctly inserted in the next block. For all $(\ell, d) \in \mathcal{L} \times \mathsf{D}$ and $(a, z) \in \Sigma \times \mathsf{D}$ in the same block:
5. If $d = z$ and there exists a transition $\ell \xrightarrow{=r,\ a} \ell'$ (with or without update), then (ℓ', d) must be in the next block.

6. If $d \neq z$ and there exists a transition $\ell \xrightarrow{\neq r,\ a} \ell'$, then (ℓ', d) must be in the next block.

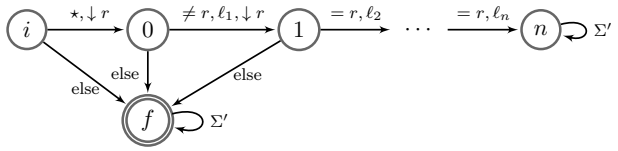
7. If $d \neq z$ and there exists a transition $\ell \xrightarrow{\neq r,\ a,\ \downarrow r} \ell'$ then (ℓ', z) must be in the next block.

By construction, we see that $\mathcal{A}$ has some synchronizing data word if and only if $\mathsf{lang} \neq \emptyset$. Below, we construct a 1-NRA $\mathcal{A}_{\mathsf{comp}}$ that accepts the complement of lang. Then, $\mathcal{A}$ has some synchronizing data word if and only if the language of $\mathcal{A}_{\mathsf{comp}}$ is not universal.

The 1-NRA $\mathcal{A}_{\mathsf{comp}}$ is the union of several 1-NRA in the family of

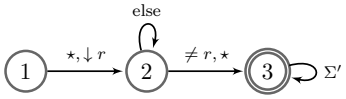
$\mathcal{A}_1, \mathcal{A}_2, \cdots, \mathcal{A}_7$ such that each family $\mathcal{A}_i$ violates the i-th condition among the membership conditions in lang.

1. Family $\mathcal{A}_1$: We add a 1-NRA that accepts data words not starting with $(\star, y)(\ell_1, x), \cdots, (\ell_n, x)$.

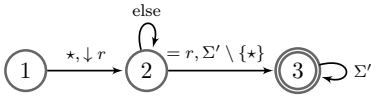


2. Family $\mathcal{A}_2$. we add a DFA that accepts data words u such that $\mathsf{proj}(u)$ is not in the regular language $(\star\,\mathcal{L}^+\Sigma)^+ \star\ \ell_{\mathsf{synch}} \star$.

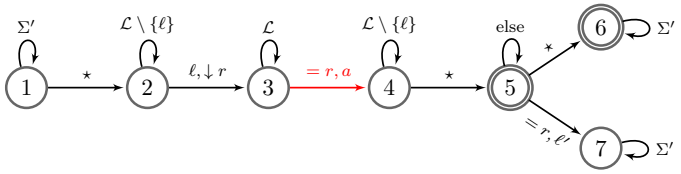
3. Family $\mathcal{A}_3$: we add a 1-NRA that accepts data words in which the two delimiters $\star$ have different data.



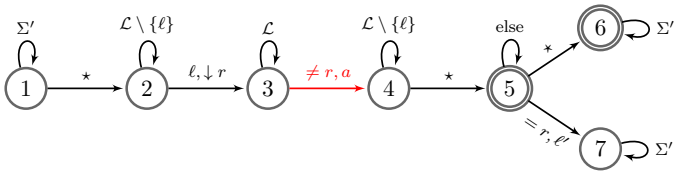
4. Family $\mathcal{A}_4$: we add a 1-NRA that accepts data words in which the datum of first $\star$ is not used only by occurrences of $\star$.



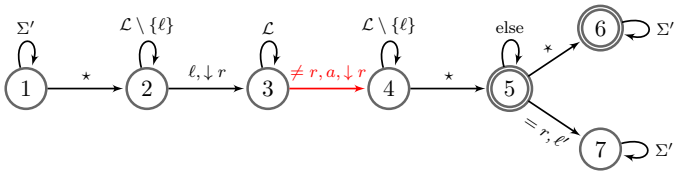
5. Family $\mathcal{A}_5$: for all transitions $\ell \xrightarrow{=r,\ a} \ell'$, we add a 1-NRA that only accepts data words such that one block contains some (ℓ, d) and (a, z) with $d = z$ where the next block does not have (ℓ', d).



6. Family $\mathcal{A}_6$: for all transitions $\ell \xrightarrow{\neq r,\ a} \ell'$, we add a 1-NRA that only accepts data words such that one block contains some (ℓ, d) and (a, z) with $d \neq z$ where the next block does not have (ℓ', d).



7. Family $\mathcal{A}_7$: for all transitions $\ell \xrightarrow{\neq r,\, a,\, \downarrow r} \ell'$, we add a 1-NRA that only accepts data words such that one block contains some (ℓ, d) and (a, z) with $d \neq z$ where the next block does not have (ℓ', z).



The proof is complete. □

Theorem 4.14. *The synchronizing problem for* 1-NRA *is* Ackermann*-complete.*

Proof. This is an immediate result of Lemmas 4.12 and 4.13 and Ackermann-completeness of the non-universality problem for 1-NRA, which follows from Theorem 2.7 and the proof of Theorem 5.2 in [36], as well as the result for incrementing counter automata in [57]. □

Conclusion and Future Work

In this thesis, we gave logical characterizations for weighted automata models on pictures and data words. Furthermore, we gave a logical characterization for Büchi-tiling systems in the spirit of the classical Büchi-Elgot-Trakhtenbrot theorem. From the theoretical point of view, our results concerning logical characterizations show the robustness of the automata-theoretic approach. In addition, it shows the effectiveness of translation processes from formulas into automata. As a more practical problem, we also considered synchronizing data words in register automata, which could be motivated by applications in planning, control of discrete systems, biocomputing, and robotics [14, 101, 40]. Below, we briefly summarize the contents of this thesis and mention some directions for the future work.

In Chapter 1, we defined weighted 2-dimensional on-line tessellation automata (W2OTA) taking weights from a new weight structure called picture valuation monoids, which are more general than semirings. The behavior of such weighted automata could be used to model, for example, average density of pictures. We proved a Nivat's theorem for W2OTA. This result provides a connection between the behaviors of W2OTA and 2OTA (2-dimensional on-line tessellation automata without weights). Indeed, we showed that recognizable picture series can be obtained precisely as projections of particularly simple unambiguously recognizable series restricted to unambiguous recognizable picture languages. In addition, we showed that if we consider idempotent picture valuation monoids, then we obtain this result without unambiguity condition. In addition, we defined a new weighted monadic second-order logic (WMSO), and as the second main result of this chapter, we proved that under certain assumptions on the underlying picture valuation monoid, our W2OTA and suitable fragments of our WMSO are expressively equivalent.

In Chapter 2, we introduced the notions of Büchi-tiling systems and Büchi-tiling recognizable $+\omega$-languages. We showed that the class of all

Büchi-tiling recognizable $+\omega$-picture languages has the similar closure properties as the class of tiling recognizable languages of finite pictures: it is closed under projection, union, and intersection, but not under complementation. While for languages of *finite* pictures, tiling recognizability and EMSO-definability coincide [64], the situation is quite different for languages of $+\omega$-pictures: In this setting, the notion of tiling recognizability does not even cover the language of all $+\omega$-pictures over $\Sigma = \{a, b\}$ in which the letter a occurs at least once – a picture-language that can easily be defined in first-order logic. As a consequence, EMSO is too strong for being captured by the class of tiling recognizable $+\omega$-picture languages. On the other hand, EMSO is too weak for being captured by the class of all *Büchi*-tiling recognizable $+\omega$-picture languages. To obtain a logical characterization of this class, we introduced the logic EMSO^∞, which extends EMSO with existential quantification of *infinite* sets. Additionally, using combinatorial arguments, we showed that the Büchi characterization theorem for ω-regular languges does not carry over to the Büchi-tiling recognizable $+\omega$-picture languages. Concerning future work, a generalization of the results in this chapter to the quantitative setting would be interesting.

In Chapter 3, we introduced a model of weighted register automata and we gave an expressively equivalent weighted logic. On the one hand, our results showed the robustness of the automata-theoretic approach, helped to understand better the behaviors of weighted register automata and made it possible to incorporate timed automata [2] and weighted timed automata [5, 86] into our framework. On the other hand, our expressive equivalence result could be used as a basis for the quantitative verification of systems with data, e.g., for the study of quantitative extensions of temporal logics on data words [36]. An important open question concerns algorithmic properties of weighted register automata. We believe that the optimal reachability problem for weighted register automata is decidable for various examples considered in this paper. It could be interesting to extend our results to the setting of infinite data words and data trees and to investigate, in the setting of data words, the cases where the weight measure cannot be modelled using semirings (e.g., average or discounted costs, energy problems and weighted register automata with multiple cost parameters). Note that these nonclassical weight measures have been extensively studied in the setting of weighted timed automata. It could be also interesting to compare the expressive power of our register automata model with the data automata model of [22]. We believe that they are incomparable. An extension of class register automata and the logic captured by them [18],

where data words have been considered as behavioral models of concurrent systems, to the weighted setting could be attractive, as well.

Another interesting open problem is to find a connection between the two structures considered in this thesis, namely pictures and data words. This problem could lead us to define the class of data picture languages and extend the existing results for data words to two dimensions. Data picture languages could be motivated by problems arising from image processing, as well. By a data picture, we mean a rectangular array of pairs where the first element is taken from a finite alphabet and the second element is taken from an infinite data domain. A new automaton model equipped with a finite set of registers which can operate on data pictures in two dimensions, could be used for several applications such as image comparison, e.g. finding the similarity of an image with other images from a set.

In Chapter 4, we studied the concept of synchronizing data words in register automata over the data structure $\mathbb{D} = (\mathsf{D}, \{=_{\mathsf{D}}, \neq_{\mathsf{D}}\})$. Synchronizing problem for data words asks whether there exists a data word that sends all states of the register automaton to a single state. We provided the complexity bounds of the synchronizing problem in the family of deterministic register automata with k registers (k-DRA), and in the family of nondeterministic register automata with single register (1-NRA), and in general undecidability of the problem in the family of k-NRA. To this aim, we proved that, for DRA, inputting data words with only $2k + 1$ distinct data values, from the infinite data domain, is sufficient to synchronize. Then, we showed that the synchronizing problem for DRA is in general PSPACE-complete, and it is in NLOGSPACE for 1-DRA. For nondeterministic register automata (NRA), we showed that Ackermann(n) distinct data, where n is the number of states of the register automaton, might be necessary to synchronize. Then, by means of a construction, proving that the synchronizing problem and the non-universality problem in 1-NRA are interreducible, we could show the Ackermann-completeness of the problem for 1-NRA. However, for k-NRA, in general, we proved that this problem is undecidable due to the unbounded length of synchronizing data words. Now an important open question concerns the bounded synchronizing problem, which requires the synchronizing data words to have at most a given length. As another open problem in this setting one could also consider the synchronizing problem in register automata over more general data structures than the one considered in this chapter.

Bibliography

[1] J. H. Altenbernd, W. Thomas, S. Wöhrle: Tiling systems over infinite pictures and their acceptance conditions. In Proceedings of the 6th International Conference on Developments in Language Theory (DLT 2002), Lecture Notes in Computer Science, vol. 2450, 297-306, Springer, 2002.

[2] R. Alur , D. L. Dill: A theory of timed automata. Theoretical Computer Science, vol. 126, no. 2, 183-235, 1994.

[3] R. Alur, L. Fix, T. A. Henzinger: Event-clock automata: a determinizable class of timed automata. Theoretical Computer Science, vol. 211(1-2), 253-273, 1999.

[4] R. Alur, P. Madhusudan: Visibly pushdown languages. In Proceedings of the 36th Annual ACM Symposium on Theory of Computing (STOC 2004), 202-211. Association for Computing Machinery, 2004.

[5] R. Alur, S. La Torre, G. J. Pappas: Optimal paths in weighted timed automata. In Proceedings of the 4th International Workshop on Hybrid Systems: Computation and Control (HSCC 2001), Lecture Notes in Computer Science, vol. 2034, 49-62, Springer, 2001.

[6] R. Angles, C. Gutierrez: Survey of graph database models. ACM Computing Surveys, 40, no. 1, 1-39, 2008.

[7] M. Anselmo, D. Giammarresi, M. Madonia, and A. Restivo: Unambiguous recognizable two-dimensional languages. Theoretical Information and Application, vol. 40, no. 2, 277-293, 2006.

[8] P. Babari, M. Droste: A Nivat theorem for weighted picture automata and weighted MSO logics. In Proceedings of the 9th International Conference on Language and Automata: Theory and Applications (LATA 2015), Lecture Notes in Computer Science, vol. 8977, 703-715, Springer, 2015.

[9] P. Babari, M. Droste, V. Perevoshchikov: Weighted register automata and weighted logic for data words. In Proceedings of the 13th International Colloquium on Theoretical Aspects of Computing (ICTAC 2016), Lecture Notes in Computer Science, to appear.

[10] P. Babari, K. Quaas, M. Shirmohammadi: Synchronizing data words for register automata. In Proceedings of the 41st International Symposium on Mathematical Foundations of Computer Science (MFCS 2016), Leibniz International Proceedings in Informatics, vol. 58, no. 15, 1-15, Dagstuhl-Leibniz-Zentrum für Informatik, 2016.

[11] P. Babari, N. Schweikardt: $+\omega$-Picture languages recognizable by Büchi-tiling systems. In Proceedings of the 10th International Conference on Language and Automata: Theory and Applications (LATA 2016), Lecture Notes in Computer Science, vol. 9618, 76-88, Springer, 2016.

[12] P. Barceló, L. Libkin, A. W. Lin, P. T. Wood: Expressive languages for path queries over graph-structured data. ACM Transactions on Database Systems, vol. 37, no. 4, 1-46, 2012.

[13] G. Behrmann, A. Fehnker, T. Hune, K. G. Larsen, P. Petterson, J. Romijn, F. Vaandrager: Minimum-cost reachability for priced timed automata. In Proceedings of the 4th International Workshop on Hybrid Systems: Computation and Control (HSCC 2001), Lecture Notes in Computer Science, vol. 2034, 147-161, Springer, 2001.

[14] Y. Benenson, R. Adar, T. P. Elizur, Z. Livneh, E. Shapiro: DNA molecule provides a computing machine with both data and fuel. In Proceedings of the National Academy of Sciences of the United States of America, vol. 100, 2191-2196, 2003.

[15] J. Berstel: Transductions and ontext-free languages. Teubner Studienbücher. Informatik, Teubner, Stuttgart, 1979.

[16] M. Bojańczyk, C. David, A. Muscholl, T. Schwentick, L. Segoufin: Two-variable logic on data words. ACM Transactions on Computational Logic, vol. 12, no. 4, 27:1- 27:26, 2011.

[17] M. Bojańczyk, P. Parys: Xpath evaluation in linear time. Association for Computing Machinery, vol. 58, no.4, 17:1-17:33, 2011.

[18] B. Bollig: An automaton over data words that captures EMSO logic. In Proceedings of the 22nd International Conference on Concurrency

Theory (CONCUR 2011), Lecture Notes in Computer Science, vol. 6901, 71-186, Springer, 2011.

[19] B. Bollig, P. Gastin: Weighted versus probabilistic logic. In Proceedings of the 13th International Conference on Developments in Language Theory (DLT 2009), Lecture Notes in Computer Science, vol. 5583, 18-38, Springer, 2009.

[20] B. Bollig, D. Kuske: Muller message-passing automata and logics. Information and Computation, vol 206, no. 9-10, 1084-1094, 2008.

[21] A. Bouajjani, P. Habermehl, Y. Jurski, M. Sighireanu: Rewriting systems with data. In Proceedings of the 16th International Symposium on Fundamentals of Computation Theory (FCT 2007), Lecture Notes in Computer Science, vol. 4639, 1-22, Springer, 2007.

[22] P. Bouyer: A logical characterization of data languages. Information Processing Letters, vol. 84, no. 2, 75-85, 2002.

[23] P. Bouyer, A. Petit, D. Thérien: An algebraic characterization of data and timed languages. In Proceedings of the 12th International Conference on Concurrency Theory (CONCUR 2001), Lecture Notes in Computer Science, vol. 2154, 248-261. Springer, 2001.

[24] S. Bozapalidis, A. Grammatikopoulou: Recognizable picture series. Automata, Languages and Combinatorics, vol. 10, 159-183, 2005.

[25] J. R. Büchi: Weak second order arithmetic and finite automata. Zeitschrift für Mathematische Logik und Grundlagen der Informatik, vol. 6, 66-92, 1960.

[26] J. Ćerný: Poznámka k homogénnym experimentom s konečnými automatmi. Matematicko-fyzikálny časopis, vol. 14, no. 3, 208-216, 1964.

[27] K. Chatterjee, L. Doyen: Computation tree logic for synchronization properties. In Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016), Leibniz International Proceedings in Informatics, to appear.

[28] K. Chatterjee, L. Doyen, T. A. Henzinger: Alternating weighted automata. In Proceedings of the 18th International Symposium on Fundamentals of Computation Theory (FCT 2009), Lecture Notes in Computer Science, vol. 5699, 3 -13, Springer, 2009.

[29] K. Chatterjee, L. Doyen, T. A. Henzinger: Expressiveness and closure properties for quantitative languages. Logical Methods in Computer Science, vol. 6(3-10), 1-23, 2010.

[30] K. Chatterjee, L. Doyen, T. A. Henzinger: Probabilistic weighted automata. In Proceedings of the 20th International Conference on Concurrency Theory (CONCUR 2009), Lecture Notes in Computer Science, vol. 5710, 244-258, Springer, 2009.

[31] K. Chatterjee, L. Doyen, T. A. Henzinger: Quantitative languages. ACM Transactions on Computational Logic, vol. 11, no. 4, 23:1-23:38, 2010.

[32] D. Chistikov, P. Martyugin, M. Shirmohammadi: Synchronizing automata over nested words. In Proceedings of the 19th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS 2016), Lecture Notes in Computer Science, vol. 9634, 252-268., Springer, 2016.

[33] L. Clemente, S. Lasota: Timed pushdown automata revisited. In Proceedings of the 30th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2015), 738-749, 2015.

[34] T. Colcombet, C. Ley, G. Puppies: On the use of guards for logics with data. In Proceedings of the 36th International Symposium on Mathematical Foundations of Computer Science (MFCS 2011), Lecture Notes in Computer Science, vol. 6907, 243-255, Springer, 2011.

[35] V. R. Dare, K. G. Subramanian, D. G. Thomas, R. Siromoney: Infinite arrays and recognizability. International Journal of Pattern Recognition and Artificial Intelligence, vol. 14, 525-536, 2000.

[36] S. Demri, R. Lazić: LTL with the freeze quantifier and register automata. International Journal of ACM Transactions on Computational Logic (TOCL), vol. 10, no. 3, 16:1-16:30, 2009.

[37] S. Demri, R. Lazic, D. Nowak: On the freeze quantifier in constraint LTL: decidability and complexity. Information and Computation, vol. 205, no. 1, 2-24, 2007.

[38] S. Demri, R. Lazic, A. Sangnier: Model checking memoryful linear-time logics over one-counter automata. Theoretical Computer Science, vol. 411 (22-24), 2298-2316, 2010.

[39] L. Doyen, L. Juhl, K. G. Larsen, N. Markey, M. Shirmohammadi: Synchronizing words for weighted and timed automata. In Proceedings of the 34th Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2014), Leibniz International Proceedings in Informatics, vol. 29, 121-132, 2014.

[40] L. Doyen, T. Massart, M. Shirmohammadi: Infinite synchronizing words for probabilistic automata. In Proceedings of the 36th International Symposium on Mathematical Foundations of Computer Science (MFCS 2011), Lecture Notes in Computer Science, vol. 6907, 278-289, Springer, 2011.

[41] M. Droste, P. Gastin: Weighted automata and weighted logics. Theoretical Computer Science, vol. 380 (1-2), 69-86, 2007.

[42] M. Droste, W. Kuich, H. Vogler. (eds.): Handbook of Weighted Automata. EATCS Monographs on Theoretical Computer Science, Springer, 2009.

[43] M. Droste, D. Kuske: Weighted automata. In: Pin, J.-E. (ed.) Handbook: "Automata: from Mathematics to Applications". European Mathematical Society (to appear).

[44] M. Droste, I. Meinecke: Weighted automata and weighted MSO logics for average and longtime-behaviors. Information and Computation, vol. 220-221, 44-59, 2012.

[45] M. Droste, V. Perevoshchikov: A Nivat Theorem for weighted timed automata and weighted relative distance logic. In Proceedings of the 41st International Colloquium on Automata, Languages, and Programming (ICALP 2014), Lecture Notes in Computer Science, vol. 8573, 171-182, 2014.

[46] M. Droste, G. Rahonis: Weighted automata and weighted logics on infinite words. In Proceedings of the 10th International Conference on Developments in Language Theory (DLT 2006), Lecture Notes in Computer Science, vol. 4036, 49-58, Springer, 2006.

[47] M. Droste, H. Vogler: Weighted automata and multi-valued logics over arbitrary bounded lattices. Theoretical Computer Science, vol. 418, 14-36, 2012.

[48] M. Droste, H. Vogler: Weighted tree automata and weighted logics. Theoretical Computer Science, vol. 366, 228-247, 2006.

[49] H. D. Ebbinghaus, J. Flum: Finite Model Theory. Springer, 1995.

[50] S. Eilenberg: Automata, Languages, and Machines, vol. A. Academic Press, 1974.

[51] C. C. Elgot: Decision problems of finite automata design and related arithmetics. Transactions of the American Mathematical Society, vol. 98, 21-51, 1961.

[52] I. Fichtner: Weighted picture automata and weighted logics. Theory of Computing Systems, vol. 48, no. 1, 48-78, 2011.

[53] I. Fichtner: Characterizations of recognizable picture series. Theoretical Computer Science, vol. 374, 214-228, 2007.

[54] I. Fichtner: Weighted picture automata and weighted logics. In Proceedings of the 23rd International Symposium on Theoretical Aspects of Computer Science (STACS 2006), Lecture Notes in Computer Science, vol. 3884, 313-324, Springer, 2006.

[55] D. Figueira: Alternating register automata on finite data words and trees. Logical Methods in Computer Science, vol. 8 (1-22), 1-43, 2012.

[56] D. Figueira: Satisfiability of downward Xpath with data equality tests. In: Proceedings of the Twenty-eighth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, 197-206, 2009.

[57] D. Figueira, S. Figueira, S. Schmitz, P. Schnoebelen: Ackermannian and primitive-recursive bounds with Dickson's lemma. In Proceedings of the 26th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2011), 269-278, 2011.

[58] D. Figueira, P. Hofman, S. Lasota: Relating timed and register automata. In: EXPRESS 2010. Electronic Proceedings in Theoretical Computer Science, vol. 41, 61-75, 2010.

[59] O. Finkel: On recognizable languages of infinite pictures. International Journal of Foundations of Computer Science, vol. 15, no. 6, 823-840, 2004.

[60] K. S. Fu: Syntactic Methods in Pattern Recognition. Academic Press, New York, 1974.

[61] D. Giammarresi, A. Restivo: Recognizable picture languages. International Journal of Pattern Recognition and Artificial Intelligence, vol. 6 (2-3), 241-256, 1992.

[62] D. Giammarresi, A. Restivo: Two-dimensional finite state recognizability. Fundamental Informaticae, vol. 25, no. 3, 399-422, 1996.

[63] D. Giammarresi and A. Restivo: Two-dimensional languages. In G. Rozenberg and A. Salomaa. (eds.): Handbook of Formal Languages, vol.3, 215-267, Springer, 1997.

[64] D. Giammarresi, A. Restivo, S. Seibert, W. Thomas: Monadic second-order logic over rectangular pictures and recognizability by tiling systems. Information and Computation, vol. 125, no. 1, 32-45, 1996.

[65] S. Gnanasekaran, V. R. Dare: Infinite arrays and domino systems. Electronic Notes in Discrete Mathematics, vol. 12, 349-359, 2003.

[66] S. Gnanasekaran, V. R. Dare: On recognizable infinite array languages. In Proceedings of the 10th International Workshop on Combinatorial Image Analysis (IWCIA 2004), Lecture Notes in Computer Science, vol. 3322, 209-218, 2005.

[67] K. Inoue, A. Nakamura: Some properties of two-dimensional on-line tessellation acceptors. Information Sciences, vol. 13, 95-121, 1977.

[68] K. Inoue, A. Nakamura: Nonclosure properties of two-dimensional on-line tessellation acceptors and one-way parallel/sequential array acceptors. Transaction of IECE of Japan, vol. 6, 475-476, 1977.

[69] K. Inoue, I. Takanami: A survey of two-dimensional automata theory. Information Sciences, vol. 55, 99-121, 1991.

[70] K. Inoue, I. Takanami: A characterization of recognizable picture languages. In Proceedings of the 2nd International Conference on Parallel Image Analysis (ICPIA 1992), Lecture Notes in Computer Science, vol. 654, 133-143, Springer, Berlin, 1992.

[71] M. Kaminski, N. Francez: Finite-memory automata. Theoretical Computer Science, vol. 134, no. 2, 329-363, 1994.

[72] J. Kretínský, K. G. Larsen, S. Laursen, J. Srba: Polynomial time decidability of weighted synchronization under partial observability. In Proceedings of the 26th International Conference on Concurrency Theory (CONCUR 2015), Leibniz International Proceedings in Informatics, vol. 42, 142-154, 2015.

[73] W. Kuich, A. Salomaa: Semirings, Automata, Languages, vol. 6 of EATCS Monographs on Theoretical Computer Science. Springer, 1986.

[74] K. G. Larsen, S. Laursen, J. Srba: Synchronizing strategies under partial observability. In Proceedings of the 25th International Conference on Concurrency Theory (CONCUR 2014), Lecture Notes in Computer Science, vol. 8704, 188-202, Springer, 2014.

[75] M. Latteux, D. Simplot: Recognizable picture languages and domino tiling. Theoretical Computer Science, vol. 178, 275-283, 1997.

[76] K. Lindgren, C. Moore, and M. Nordahl: Complexity of two-dimensional patterns. Statistical Physics, vol. 91 (5-6), 909-951, 1998.

[77] A. Lisitsa, I. Potapov: Temporal logic with predicate lambda-abstraction. In Proceedings of the 12th International Symposium on Temporal Representation and Reasoning (TIME 2005), 147-155. IEEE Computer Society, 2005.

[78] P. V. Martyugin: Complexity of problems concerning carefully synchronizing words for PFA and directing words for NFA. In Proceedings of the 5th International Computer Science Symposium in Russia (CSR 2010), Lecture Notes in Computer Science, vol. 6072, 288-302, Springer, 2010.

[79] O. Matz: On piecewise testable, starfree, and recognizable picture languages. In Proceedings of the 1st International Conference on Foundations of Software Science and Computation Structures (FoSSaCS 1998), Lecture Notes in Computer Science, vol. 1378, 203-210. Springer, 1998.

[80] I. Meinecke: Weighted logics for traces. In Proceedings of the 1st International Computer Science Symposium in Russia (CSR 2006), Lecture Notes in Computer Science, vol. 3967, 235-246, 2006.

[81] M. Minski, S. Papert: Perceptron: An Introduction to Computational Geometry. M.I.T. Press, Cambridge, Mass. 1969.

[82] F. Neven, T. Schwentick, V. Vianu: Finite state machines for strings over infinite alphabets. ACM Transactions on Computational Logic, vol. 5, no. 3, 403-435, 2004.

[83] M. Nivat: Transductions des langages de Chomsky. In: Annales Institut Fourier, vol. 18, no. 1, 339-456, 1968.

[84] J. E. Pin: Sur les mots synthronisants dans un automate fini. Elektronische Informationsverarbeitung und Kybernetik, vol. 14, no. 6, 297-303, 1978.

[85] K. Quaas: Kleene-Schützenberger and Büchi theorems for weighted timed automata. PhD thesis, Universität Leipzig, 2010.

[86] K. Quaas: MSO logics for weighted timed automata. Formal Methods in System Design, vol. 38, no. 3, 193-222, 2011.

[87] H. Sakamoto, D. Ikeda: Intractability of decision problems for finite-memory automata. Theoretical Computer Science, vol. 231, no. 2, 297-308, 2000.

[88] A. Salomaa, M. Soittola: Automata-Theoretic Aspects of Formal Power Series. Texts and Monographs on Computer Science, Springer, 1978.

[89] S. Schmitz: Complexity hierarchies beyond elementary. ACM Transactions on Computation Theory, vol. 8, no. 1, 3:1-3:36, 2016.

[90] M. P. Schützenberger: On the definition of a family of automata. Information and Control, vol. 4, 245-270, 1961.

[91] M. Shirmohammadi: Qualitative analysis of probabilistic synchronizing systems. Phd thesis, Université Libre de Bruxelles, and Ecole Normale Supérieure de Cachan, 2014.

[92] D. Simplot: A characterization of recognizable picture languages by tilings by finite sets. Theoretical Computer Science, 218, no. 2, 297-323, 1999.

[93] R.A. Smith: Two-dimensional formal languages and pattern recognition by cellular automata. In Proceedings of the 12th Annual Symposium on Switching and Automata Theory (FOCS 1971), 144-152, IEEE Computer Society, 1971.

[94] L. J. Stockmeyer, A. R. Meyer: Word problems requiring exponential time. In Proceedings of the 5th Annual ACM Symposium on Theory of Computing (STOC 1973), 1-9, Association for Computing Machinery, 1973.

[95] K. G. Subramanian, K. Rangarajan, M. Mukund. Formal Models: Languages and Applications. Series in Machine Perception and Artificial Intelligence, vol. 66, 2006.

[96] J. W. Thatcher, B. Wright: Generalized finite automata theory with an application to a decision problem of second-order logic. Mathematical System Theory, vol. 2, 57-81, 1968.

[97] W. Thomas: Languages, Automata, and Logic. Handbook of formal languages, vol. 3, 389-455, Springer, 1997.

[98] W. Thomas: On logics, tilings, and automata. In Proceedings of the 18th International Colloquium on Automata, Languages, and Programming (ICALP 1991), Lecture Notes in Computer Science, vol. 510, 441-453, Springer, 1991.

[99] N. Tzevelekos: Fresh-register automata. In Proceedings of the 38th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL 2011), vol. 46, 295-306, 2011.

[100] T. Wilke: Star-free picture expressions are strictly weaker than first-order logic. In Proceedings of the 24th International Colloquium on Automata, Languages, and Programming (ICALP 1997), Lecture Notes in Computer Science, vol. 1256, 347-357, Springer, 1997.

[101] M. V. Volkov: Synchronizing automata and the cerny conjecture. In Proceedings of the 2nd International Conference on Language and Automata Theory and Applications (LATA 2008), Lecture Notes in Computer Science, vol. 5196, 11-27, Springer, 2008.

[102] P. S. P. Wang. Array Grammars, Patterns and Recognizers. World Scientific Series in Computer Science, vol. 18, 1989.

[103] T. Wilke: Automaten und Logiken zur Beschreibung zeitabhängiger Systeme. PhD thesis, Christian-Albrecht-Universität Kiel, 1994.

[104] T. Wilke: Specifying timed state sequences in powerful decidable logics and timed automata. In Proceedings of the 3rd International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems (FTRTFT 1994), Lecture Notes in Computer Science, vol. 863, 694-715, Springer, 1994.

List of Figures

List of Tables

Daten zur Autorin

Name:	Parvaneh Babari
Geburtsdatum:	21.06.1985
9/2004 - 7/2008	Bachelor Studium der Mathematik Universität Mazandaran, Babolsar, Iran
9/2008 - 1/2011	Master Studium der Mathematik Universität Tarbiat Modares, Teheran, Iran
12/2012 - 9/2016	Doktorandin am Institut für Informatik Leipzig

Bibliographische Daten

Quantitative Automata and Logic for Pictures and Data Words
(Quantitative Automaten und Logik für Bilder und Datenworte)
Babari, Parvaneh
Universität Leipzig, Dissertation, 2016
167 Seiten, 14 Abbildungen, 104 Referenzen

Selbstständigkeitserklärung

Hiermit erkläre ich, die vorliegende Dissertation selbständig und ohne unzulässige fremde Hilfe angefertigt zu haben. Ich habe keine anderen als die angeführten Quellen und Hilfsmittel benutzt und sämtliche Textstellen, die wörtlich oder sinngemäß aus veröffentlichten oder unveröffentlichten Schriften entnommen wurden, und alle Angaben, die auf mündlichen Auskünften beruhen, als solche kenntlich gemacht. Ebenfalls sind alle von anderen Personen bereitgestellten Materialien oder erbrachten Dienstleistungen als solche gekennzeichnet.

Leipzig, den 9. September 2016

. . . . . .
(Parvaneh Babari)